

The Edward S. Rogers Sr. Dept of Electrical and Computer Engineering  
University of Toronto

ECE496Y Design Project Course - Final Report

Title: **An optical wireless USB interface** (Team 3)

Project I.D. # **1482001**

Prepared by:

Kai-chung Richard Hou, [richard.hou@utoronto.ca](mailto:richard.hou@utoronto.ca)  
Eric Hsiao, [eric.hsiao@utoronto.ca](mailto:eric.hsiao@utoronto.ca)  
Roger Hung, [roger\\_hung@pchome.com.tw](mailto:roger_hung@pchome.com.tw)

Supervisor:

Khoman Phang

Section #:

5

Section  
Coordinator:

Ross Gillett

Date:

April 12, 2002

# TABLE OF CONTENTS

<b>TABLE OF CONTENTS</b>	<b>1</b>
<b>LIST OF FIGURES</b>	<b>4</b>
<b>LIST OF TABLES</b>	<b>6</b>
<b>ACKNOWLEDGEMENT</b>	<b>7</b>
<b>1 INTRODUCTION</b>	<b>8</b>
1.1 OBJECTIVE	8
1.2 MOTIVATION	8
1.2.1 UNIVERSAL SERIAL BUS	8
1.2.2 INFRARED	10
1.3 REPORT OUTLINE	12
<b>2 BACKGROUND</b>	<b>14</b>
2.1 USB SIGNALING CHARACTERISTICS	14
2.1 USB PACKETS	15
2.2 USB TRANSACTIONS	16
<b>3 USB MODULE</b>	<b>18</b>
3.1 USB TRANSCEIVER	18
3.2 DYNAMIC DEVICE DETECTION	20
<b>4 USB-IR BRIDGE</b>	<b>22</b>
4.1 MODULATOR	22
4.1.1 MODULATOR FUNCTIONS	23
4.1.2 MODULATOR IMPLEMENTATION	24
4.1.3 MODULATOR SIMULATION	25
4.2 DEMODULATOR	25
4.2.1 DEMODULATOR FUNCTIONS	25
4.2.2 DEMODULATOR IMPLEMENTATION	26
4.2.3 DEMODULATOR SIMULATION	29
4.3 DATA DIRECTION CONTROLLER	30
4.3.1 CONTROLLER DESIGN PLAN	30
4.3.2 CONTROLLER STATE MACHINE	31

<b>4.4 SAMPLER</b>	<b>33</b>
<b>4.5 OVERALL BRIDGE CIRCUIT</b>	<b>34</b>
4.5.1 CONTROL OF DATA FLOW DIRECTION	34
4.5.2 OVERALL BRIDGE SIMULATION	35
4.5.3 USB-IR BRIDGE PERFORMANCE	36
<b>5 INFRARED MODULE</b>	<b>39</b>
<b>5.1 TECHNICAL ISSUES ON INFRARED TRANSCEIVER DESIGN</b>	<b>39</b>
<b>5.2 TRANSMITTER DESIGN</b>	<b>40</b>
5.2.1 SELECTION OF LIGHT EMITTING DIODE	41
5.2.2 NEED FOR DRIVER CIRCUIT	42
5.2.3 DESIGN OF DRIVER CIRCUIT	43
5.2.4 DECOUPLING CAPACITOR	44
5.2.5 DRIVER CIRCUIT TESTING	44
<b>5.3 INFRARED RECEIVER CIRCUIT DESIGN</b>	<b>47</b>
5.3.1 SELECTION OF PHOTO DETECTOR:	47
5.3.2 PHOTOCURRENT WITH VARYING DISTANCE	48
5.3.3 PROPOSED BER AND SNR	49
5.3.4 SELECTION OF RESISTANCE AT FIRST STAGE	52
5.3.5 SECOND STAGE: PREAMPLIFIER	53
5.3.6 THIRD STAGE: COMPARATOR	53
5.3.7 TESTING CIRCUIT WITH A FIXED COMPARATOR INPUT	53
5.3.8 DC BLOCKER: COMPENSATION FOR DYNAMIC THRESHOLD	56
5.3.9 CLAMPING CIRCUITS: TO DETERMINE A PRECISE THRESHOLD VALUE	59
5.3.10 DESIGN OF CLAMPING CIRCUIT	59
5.3.11 TESTING CLAMPING CIRCUITS	60
5.3.12 OVERALL RECEIVER SYSTEM	63
<b>6 SYSTEM IMPLEMENTATION</b>	<b>64</b>
<b>6.1 BREADBOARD</b>	<b>64</b>
<b>6.2 MICRO-STRIP CIRCUIT BOARD</b>	<b>64</b>
<b>6.3 PCB (PRINTED CIRCUIT BOARD)</b>	<b>65</b>
6.3.1 DESIGNING	66
6.3.2 IMPLEMENTATION	67
6.3.3 PROCEDURES / DIRECTIONS	69
<b>7 CONCLUSION</b>	<b>74</b>
<b>7.1 COST ESTIMATE</b>	<b>74</b>
<b>7.2 FUTURE DEVELOPMENT</b>	<b>75</b>
<b>REFERENCE</b>	<b>76</b>

## **APPENDIX**

---

- A Host Modulator**
- B Host Demodulator**
- C Host Data Direction Controller**
- D Sampler for Infrared Input**
- E Original Task Assignment**
- F Revised Task Assignment**
- G Team Member Contribution**

## List of Figures

FIGURE 1 MARKET SHARE OF USB PERIPHERALS IN US RETAIL MARKET.....	9
FIGURE 2 HOST SIDE INTERFACE .....	13
FIGURE 3 DEVICE SIDE INTERFACE .....	13
FIGURE 4 USB SIGNALLING STATES.....	15
FIGURE 5 A TYPICAL USB PACKET .....	15
FIGURE 6 PACKET FORMATS .....	16
FIGURE 7 (A)TRANSACTION FROM DEVICE TO HOST, (B)TRANSACTION FROM HOST TO DEVICE .....	16
FIGURE 8 USB MODULE.....	18
FIGURE 9 USB TRANSCEIVER LOGIC DIAGRAM .....	19
FIGURE 10 DYNAMIC DEVICE DETECTION SCHEME .....	21
FIGURE 11 USB-IR BRIDGE .....	22
FIGURE 12 PULSE POSITIONING MODULATION .....	23
FIGURE 13 BASIC FUNCTION OF MODULATOR .....	24
FIGURE 14 MODULATOR SIMULATION .....	25
FIGURE 15 DEMODULATOR STATE DIAGRAM .....	27
FIGURE 16 DEMODULATOR SIMULATION .....	29
FIGURE 17 DATA FLOW CONTROLLER STATE DIAGRAM .....	31
FIGURE 18 OVERALL BRIDGE SCHEMATIC.....	35
FIGURE 19 BRIDGE SIMULATION.....	36
FIGURE 20 SIMULATION OF A DIRECT CONNECTION .....	37
FIGURE 21 TOPOLOGY OF INFRARED TRANSMITTER .....	40
FIGURE 22 INFRARED TRANSMITTER CIRCUIT .....	41
FIGURE 23 DIRECTLY FEEDING LED WITH SIGNAL .....	42
FIGURE 24 VCC OSCILLATION WITH (A) NO DECOUPLING CAPACITOR ( $V_{PP} = 5.6V$ ) (B) $C_D = 0.1\mu F$ ( $V_{PP} = 228mV$ ) (C) $C_D = 10\mu F$ ( $V_{PP} = 38mV$ ).....	44
FIGURE 25 EXPERIMENT SETUP FOR MEASURING DRIVING CURRENT.....	45
FIGURE 26 FUNCTION GENERATOR OUTPUT AT FREQUENCIES OF (A) 500KHz (B) 1MHz (C) 2MHz .....	45
FIGURE 27 OUTPUT WAVEFORM.....	46
FIGURE 28 TRANSMITTER CIRCUIT BUILT ON VECTOR BOARD .....	46
FIGURE 29 EXPERIMENT SETUP FOR MEASURING RELATIONSHIP BETWEEN PHOTO CURRENT AND DISTANCE .....	48
FIGURE 30 BER TESTING CIRCUIT .....	52
FIGURE 31 OUTPUT WAVEFORM AT THE FIRST STAGE OF RECEIVER WITH RESISTOR VALUE OF (A) 100 OHM, (B) 1000 OHM, (C) 10 KOHM.....	52
FIGURE 32 EXPERIMENT SETUP FOR TESTING RECEIVER CIRCUIT WITH FIXED THRESHOLD	54
FIGURE 33 OVERALL SYSTEM TESTING ON VECTOR BOARD.....	54
FIGURE 34 INPUT OF TRANSMITTER AND OUTPUT OF RECEIVER WITH 2.5V THRESHOLD LEVEL WHEN 5MM APART .....	55
FIGURE 35 TRANSMITTER INPUT AND RECEIVER OUTPUT WHEN THRESHOLD LEVEL(2.5V) IS (A) SMALLER THAN MEAN VALUE (2MM APART) (B) LARGER THAN MEAN (8MM APART) .....	56

FIGURE 36 EXPERIMENT SETUP FOR TESTING DC-BLOCKER .....	56
FIGURE 37 DC BLOCKER VOLTAGE WAVEFORM WHEN CAPACITOR OF VALUE (A) 4.7uF (B) 100nF (C) 330 pF (D) 10 pF .....	58
FIGURE 38 CIRCUIT SCHEMATICS FOR CLAMPING CIRCUIT WHICH OBTAIN THE MAXIMUM INPUT LEVEL .....	59
FIGURE 39 CIRCUIT SCHEMATICS FOR CLAMPING CIRCUIT WHICH OBTAIN THE MINIMUM INPUT LEVEL .....	60
FIGURE 40 TEST SETUP FOR TESTING CLAMPING MAXIMUM CLAMPING CIRCUIT .....	60
FIGURE 41 CLAMPING CIRCUIT INPUT AND OUTPUT WHEN CAPACITANCE OF (A) 10pF (B) 330pF.....	61
FIGURE 42 TEST SETUP FOR TESTING CLAMPING MAXIMUM CLAMPING CIRCUIT .....	62
FIGURE 43 OVERALL RECEIVER SYSTEM.....	63
FIGURE 45 DESIGN LAYOUT OF THE USB-IR BRIDGE CONTROLLER.....	70
FIGURE 46 DESIGN LAYOUT PHOTOCOPIED ON THE PRESS-N-PEEL FILM.....	70
FIGURE 47 USE CLOTHES IRON TO TRANSFER THE IMAGE ONTO THE BOARD .....	71
FIGURE 48 IMAGE OF THE TRACE ON THE COPPER BOARD .....	71
FIGURE 49 ETCH THE COPPER BOARD WITH FERRIC CHLORIDE.....	72
FIGURE 50 PCB AFTER ETCHING. SOCKET AND WIRES BEEN SOLDERED ON.....	72
FIGURE 51 USB-IR TEST BOARD – FIRST ATTEMPT .....	73

## List of Tables

TABLE 1 MODULATION MAP .....	25
TABLE 2 COMPARISON OF AVAILABLE INFRARED LED .....	42
TABLE 3 N-CHANNEL MOSFET SPECIFICATIONS .....	43
TABLE 4 EXPERIMENT RESULT FOR SELECTION OF CURRENT LIMITING RESISTANCE .....	44
TABLE 5 COMPARISON OF AVAILABLE PHOTODIODES AND PHOTOTRANSISTORS .....	47
TABLE 6 PHOTOCURRENT MEASUREMENT WITH VARYING DISTANCE .....	49
TABLE 7 EXPERIMENT RESULT FOR TESTING DC-BLOCKER .....	57
TABLE 8 EXPERIMENT RESULTS FOR MAXIMUM CLAMPING CIRCUIT .....	61
TABLE 9 EXPERIMENT RESULTS FOR MINIMUM CLAMPING CIRCUIT .....	62
TABLE 10 ESTIMATED COST OF OVERALL SYSTEM .....	75

## **Acknowledgement**

We especially like to thank Prof. Khoman Phang for his support and supervision of this project. He hosted weekly meeting, discussed problems and provided laboratory room and testing equipments for our team. Also, we will like to thank all the companies who generously provide us samples to experiment. Special thanks go to Texas Instrument, Agilent, Maxim-Dallas Semiconductor, Altera, and Fairchild Semiconductor for their contribution of valuable electronic components. Nobo Weng from Inventec Corporation tirelessly help us on the technical aspects of the project. Finally, we would like the thank the other two groups for their generous help as well.

# **1 Introduction**

## **1.1 Objective**

We designed, implemented, and tested a communication system that connects an USB device to a computer through an infrared link. This communication system consists of two adaptors; one is connected to the host computer and the other is connected to the USB device. This system is compliant with USB specification and will establish communication between USB devices with the host computer automatically. Data transmission and USB device manipulation are supported by this interface. The final USB-IR interface would be tested with an USB mouse.

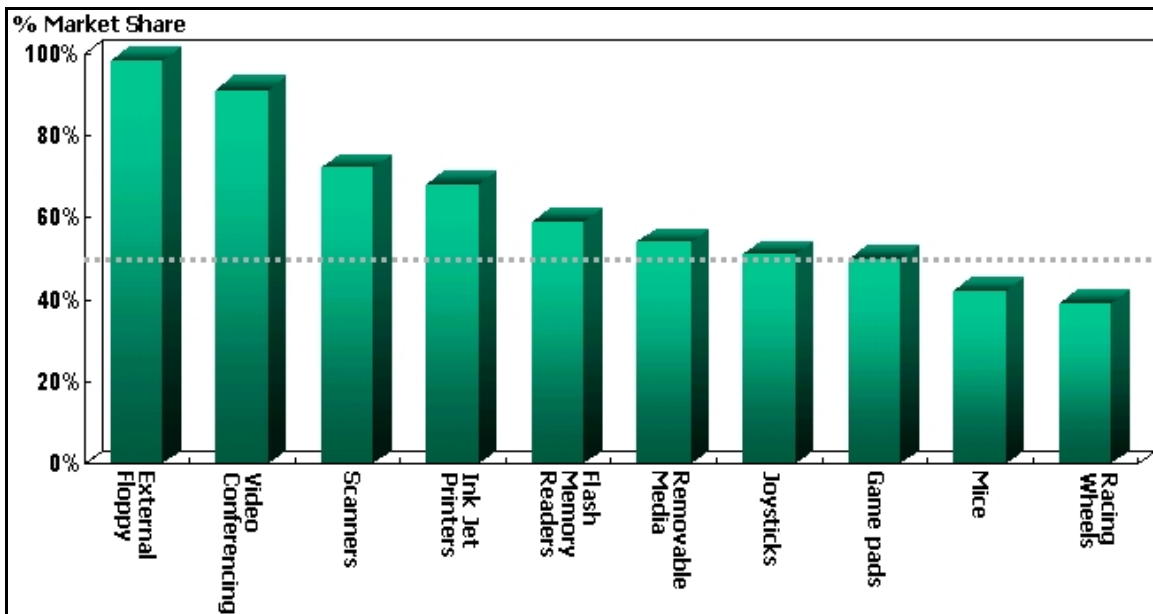
## **1.2 Motivation**

### **1.2.1 Universal Serial Bus**

Why choose to make the USB wireless? Simply because it is popular. Virtually all of the personal computers on the market today are equipped with USB ports. The number of USB-capable personal computers is projected to be 700 million by the year 2003.

Figure 1 is the market share of USB peripherals in the US retail market in the year 2000

<sup>[1]</sup>. Universal Serial Bus dominates the peripheral market.



**Figure 1 Market Share of USB Peripherals in US Retail Market**

By choosing to make such a popular standard wireless, we are making a wide range of applications wireless. The successful replacement of USB cable link with wireless link will be a major step toward a totally wireless office or home entertainment environment. The reasons for the popularity of the USB standard can be categorized as follows:

*1. The ease of use for unsophisticated end users.* The USB port has the dynamic attach/detach or “hot plug and play” capability. The feature simplifies the device installation process for end users. The USB specification also eliminates the need for the mouse and keyboard ports, and even the serial and parallel ports. This makes a much simpler PC connectivity.

*2. The ease of implementation for device developers.* The USB specification provides a more intelligent interface than the serial port standard and allows the sharing of the USB

port. This feature lowers the interface costs and thus makes a wider range of peripheral devices practical. Other features of the standards such as the simple four-pin connection, CMOS signal levels and bus power supply makes the USB specification the more attractive standard for many peripheral developers.

*3. Strong backing by industry leaders.* The USB specification models itself after the successful PCI standard. It starts out with a small group of influential industry leaders such as Intel, Microsoft and Compaq <sup>[2]</sup>. The dominance of each of the company in its field makes it possible to drive the standard into a fragmented PC market.

For the above reasons, there is a wide selection of USB devices available for the consumers. The devices range from interactive devices, such as keyboards, mice and game peripherals; to real time devices, such as telephony, digital audio and video; to portable devices, such as storage systems, cellular phones and PDA's. To make the USB wireless is to make all these devices free of the restrictions of wires.

### **1.2.2 Infrared**

The use of infrared communication can be traced back for a few decades. Over the years, the advance of technology had improved the quality of IR LED and IR photo detector. The cost of implementation with IR had been pulled down due to the high capacity of productivity and they are considered to be cheap when comparing to the Bluetooth technology, the latest and most advance short-range radio communication standard.

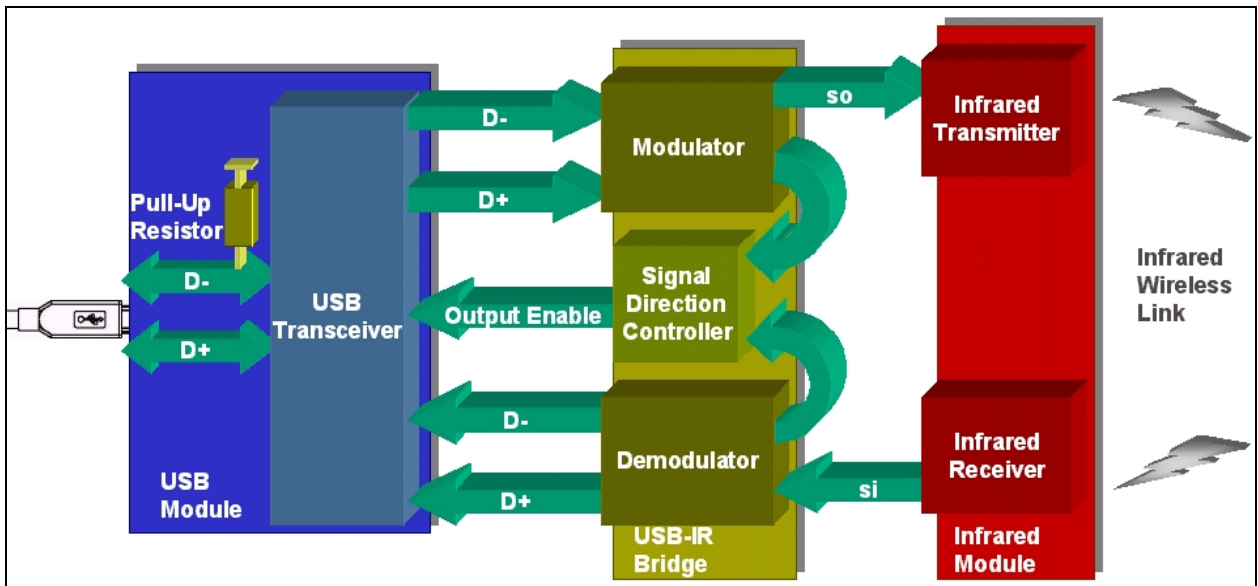
Even though IrDA is not part of this project but it still needs to be mentioned since it is the mainstream infrared communication standard. IrDA stands for Infrared Data Association. It was founded in 1993 and has been aggressively to pursue short-range optical wireless through publishing many specifications to help implementing this technology. The current fastest transfer rate is 4Mbps but they are pushing this limit to 16Mbps. About 325 millions units of mobile devices have IrDA base cumulated by end of 2000 and it is growing at a 50% annual rate. Infrared meets the basic wireless need for transferring information.

Bluetooth was the technology attracting all the attention of investors, consumers, and manufacturers for a few years. Many companies such as IBM, Toshiba, Intel... got together and structured this short-range RF communication specification in 1999 with the ambition of linking up all the electrical and electronic appliances together. The Bluetooth radio can be built into a small microchip and operates in a globally available frequency band ensuring communication compatibility worldwide. It provides effortless, wireless and instant connections between various communication devices. However, the main disadvantage for Bluetooth technology is the cost for implementing and the complexity of the system interactions. Many companies had pulled out its Bluetooth products since it is just not the time yet. The cost is still too high and most consumers don't find the need for it.

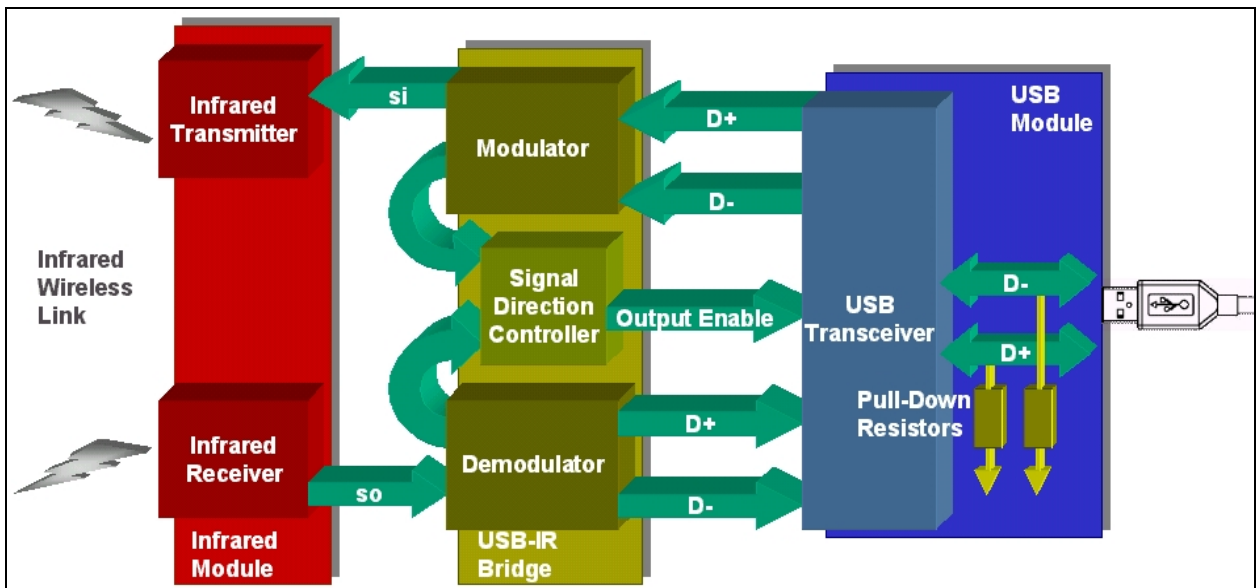
Since infrared implementation is simpler and cheaper and infrared system design is not as complex as Bluetooth, infrared is an ideal solution for the project to provide the plain wireless communication link.

### ***1.3 Report Outline***

This report includes information and different aspects of the project. The system diagrams are shown in Figure 2 and Figure 3. This report goes through the USB signaling characteristics in section 2. Then it discusses about the USB module in section 3, which contains information about the USB transceiver and device detection. Section 4 covers USB-IR bridge design and this section contains the design of the controller, modulator, demodulator and sampler and their simulation results. Next, section 5 contains all the aspects about IR transceiver and the experimental test results would be discussed. Afterward, it will discuss about method of system implementation in section 6. Then this report will be wrapped up with conclusion and future development.



**Figure 2 Host Side Interface**



**Figure 3 Device Side Interface**

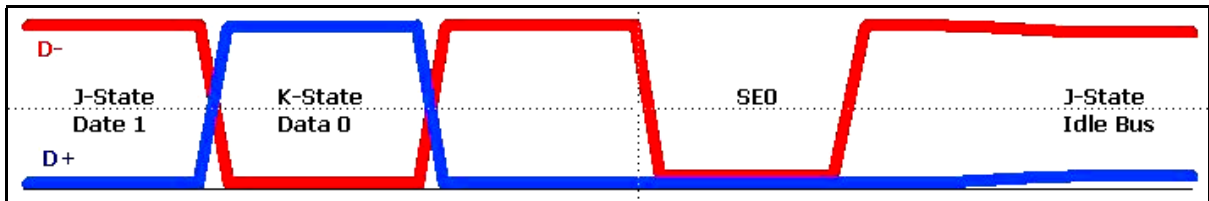
## 2 Background

The understanding of the nature of the USB data signaling is crucial for building the USB module and the bridge between the USB module and the infrared module. For the purpose of this project, we will be focusing on a simplistic USB link model, namely, an USB link between a root hub (host) and a device.

### 2.1 USB Signaling Characteristics

USB signaling is bi-directional. Data transfers from the host to the device and from the device to the host. USB signaling is half-duplex. At any given time, only one side, either the host or the device, is driving the bus. They cannot both be driving the bus at the same time <sup>[3]</sup>. There are two data rates for the USB Specification Revision 1.1: full speed at 12Mbps and low speed at 1.5Mbps <sup>[4]</sup>. To simplify our objective further, we will be focusing on the application of low speed USB devices, which has a 667ns bit time.

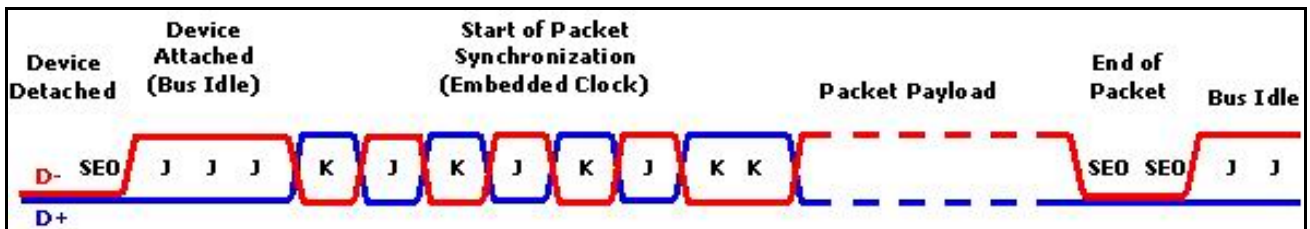
The USB transmits data by using differential signal pair D- and D+. Differential signaling has the advantages of reducing the electromagnetic interference radiation and the power dissipation of the line <sup>[5]</sup>. There are three different states for differential signals as shown in Figure 4. The *J-state* defines the idle level and the data “one” of the bus and is represented by a high D- and a low D+ at low speed. Opposite J-states are defined for low and full speed device connection <sup>[6]</sup>. The *K-state* represents a data “zero.” The K-state has the opposite differential level from J-state and is represented by a low D- and high D+ at low speed. The *Single-ended zero* or SE0 is represented by two low differential lines and is used to indicate an end of a packet or a disconnected device.



**Figure 4 USB Signalling States**

## 2.1 USB Packets

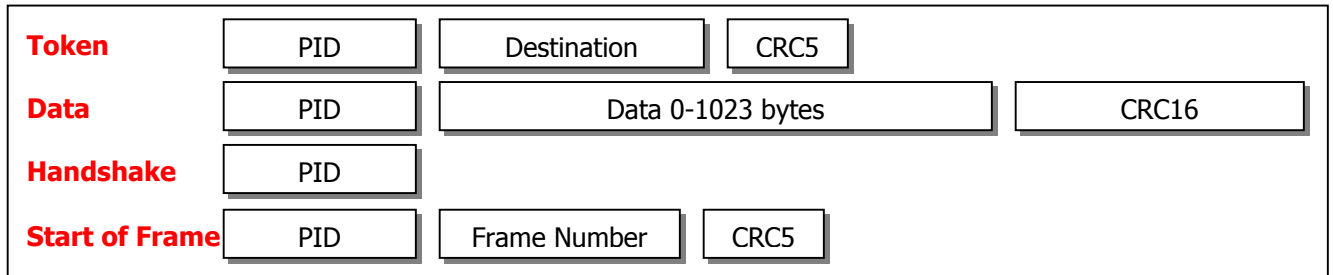
A packet is the most fundamental element of communication on the USB link. Each packet is preceded by a start-of-packet synchronization field and followed by an end-of-packet SE0 signal. A sequence of state transition from K to J (KJKJKJJK)<sup>[7]</sup> signifies the start of a packet called a synchronization field. The bus clock is embedded in this header. A typical packet including a start of packet and an end of packet is depicted in Figure 5.



**Figure 5 A Typical USB Packet**

The first byte of each packet is a packet identifier (PID). The PID is formed by four bits and the complement of itself. It identifies the type of packet that is being communicated<sup>[8]</sup>. There are four types of packets. The *token* packet is the first packet in any transaction. It specifies the transmission direction and destination. The *data* packet is the data payload packet and holds data up to 1023 bytes in size. The *handshake* packet reports the status of each transaction and is sent by the receiver to the transmitter upon the completion of each transaction. The *start of frame* packet is transmitted by the host

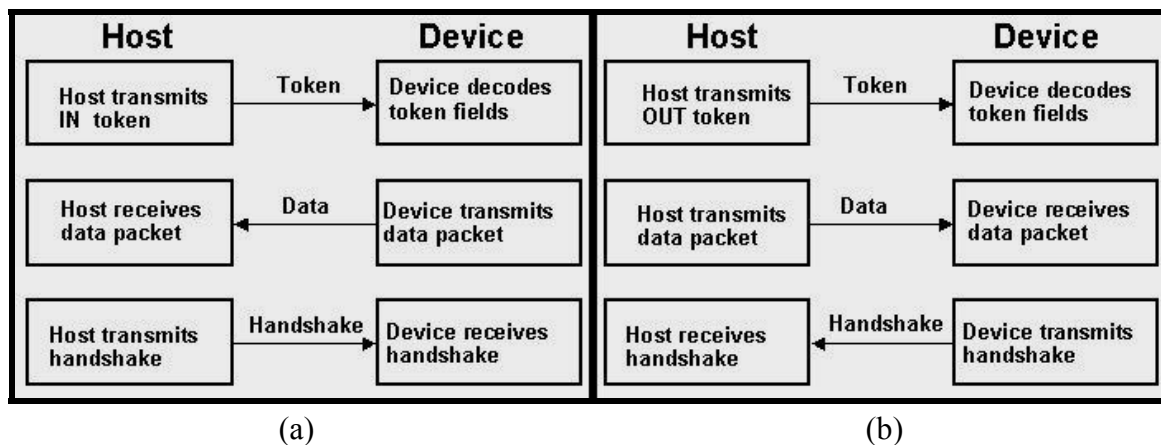
every 1ms. This is the heartbeat for any USB real-time applications. Each USB packet, except for the handshake packets, includes an error checking mechanism, named Cyclic Redundancy Check or CRC. The format of each type of packet is shown in Figure 6.



**Figure 6 Packet Formats**

## 2.2 USB Transactions

The host initiates all transactions <sup>[9]</sup>. Figure 7 (a) and (b) depict the typical transactions from a device to the host and from the host to a device respectively. The host initiates a transaction by transmitting a token packet, which specifies the direction of the transaction. The specified sender then transmits the data packet. The receiver acknowledges the transaction by sending a handshake packet.

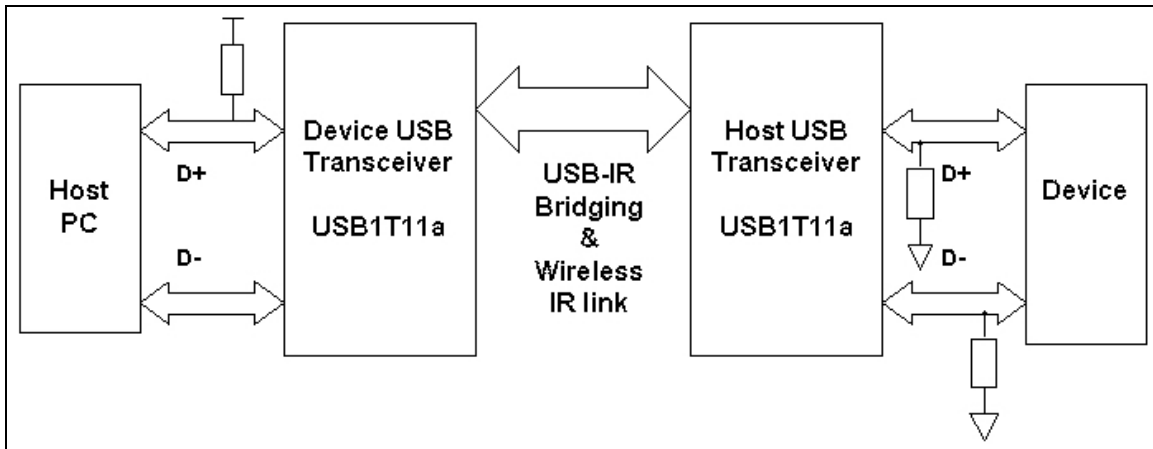


**Figure 7 (a)Transaction from Device to Host, (b)Transaction from Host to Device**

The sender of each transaction requires an immediate acknowledgement from the receiver. This is one of the main challenges of this project. The sender will wait at most 16 bit times for the acknowledgement from the receiver before it retransmits the same data packet <sup>[10]</sup>. We need to keep our overall system delay less than the required 16 bit times to avoid the situation where the sender continuously retransmits the same data. As we will see later in this report, our system has an overall delay time well below this requirement.

### 3 USB Module

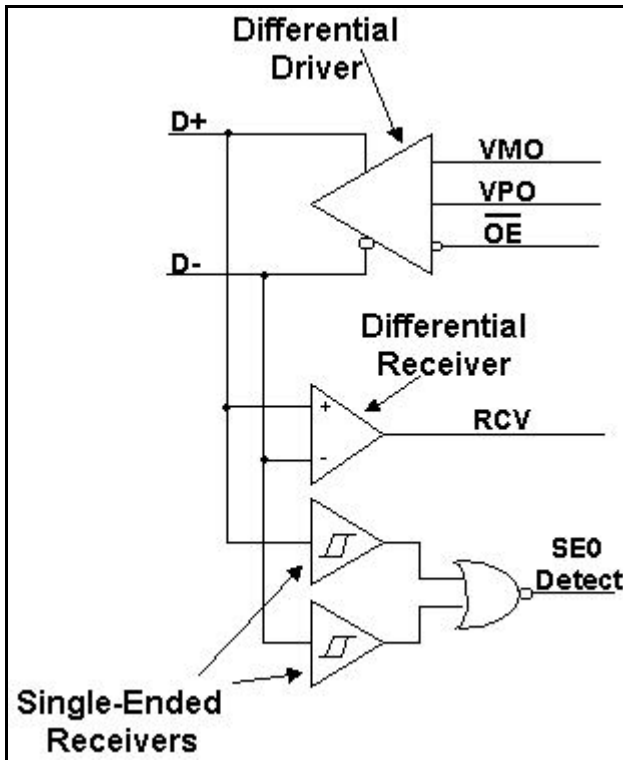
The USB module of the system has three main functions. It manages the transfer of USB differential signals, maintains the transparency of the wireless link, and preserves the dynamic device attachment capability of the USB specification. There is one USB module on each side of the wireless link. The module consists of two main components, the USB transceivers and the device detection resistors. The system configuration of the USB module is shown in Figure 8. The following subsections illustrate how each component of the USB module accomplishes the necessary functions.



**Figure 8 USB Module**

#### 3.1 USB Transceiver

USB transceivers separate the bi-directional USB bus signal into the transmission and reception components. Each transceiver consists of three components: the differential driver, the differential receiver and the single-ended receivers. The logic diagram of a USB transceiver is shown in Figure 9. The differential driver is capable of driving the three differential states, namely J-state, K-state and SE0.



**Figure 9 USB Transceiver Logic Diagram**

The output enable (OE#) pin is active low and enables VMO/VPO to drive differential signals onto the bus. The differential receiver receives signals from the bus when the OE# pin is not active. The output enable is dictated by the direction control in the USB-IR bridge. The received signal is defined as data “one” if it is higher than 3.0V and as data “zero” if it is lower than 1.0V. The receiver has sensitivity within 0.2V<sup>[11]</sup>.

The single-ended receivers serve as the detector of the SE0 state. The NOR gate produces a “one” which defines a SE0 when the differential lines D+ and D- are both low.

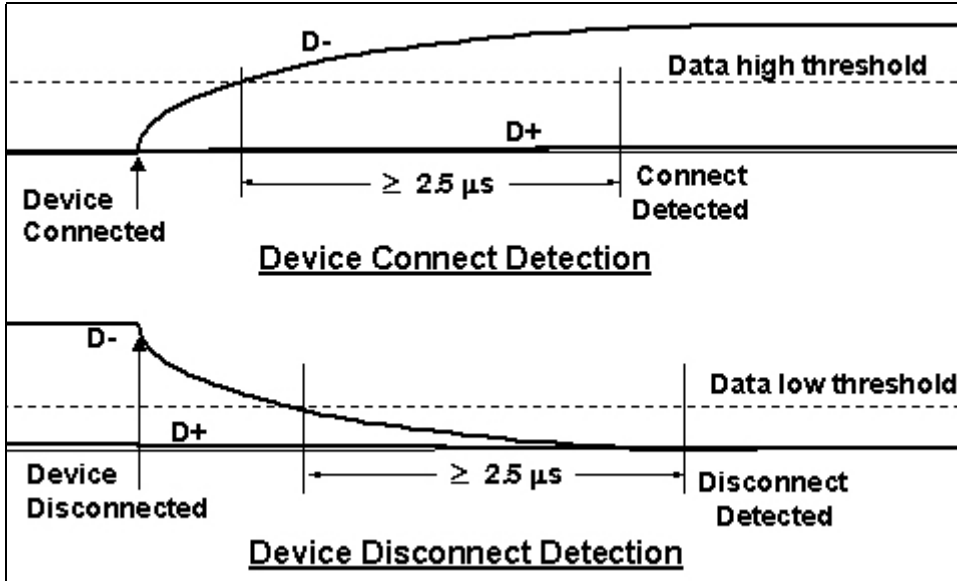
The USB transceivers are configured to maintain the transparency of the wireless link. To achieve this, it is necessary for the host and the device to have no knowledge of the wireless link. The host transceiver and the device transceiver are differentiated by their unique pull-up and pull-down resistor connection. As illustrated in Figure 8, the host is connected to the device USB transceiver, while the device is connected to the host USB transceiver. This arrangement effectively achieves a transparency of the wireless link in between the two USB transceivers.

### ***3.2 Dynamic Device Detection***

Dynamic device detection is one of the most attractive features of the USB specification. It significantly simplifies the installation procedure for peripherals. The connection or disconnection of a device is recognized by the host not through the bus power lines, but through the bus differential data lines. This feature makes it possible to preserve the dynamic device detection capability through the wireless link.

Upon the disconnection of a device, the pull-down resistors of the device USB transceiver, shown in Figure 8, pull the two differential lines D+ and D- low and form a SE0 signal. The SE0 signal of the disconnection of a device and that of the end-of-packet differ in the following manner: any SE0 state lasting longer than  $2.5\mu\text{s}$  (4 bit times) is deemed a disconnected device. An end-of-packet SE0 state has a width of two bit times. The bottom diagram of Figure 10 shows the response of the differential lines upon the disconnection of a device.

Similarly for the connection of a device, the D- differential line is driven above the data high threshold level, by the pull-up resistor, from the previous disconnection state for more than  $2.5\mu\text{s}$ , defining an idle J-state and signifying the connection of a device. The top diagram of Figure 10 shows such an event <sup>[12]</sup>.

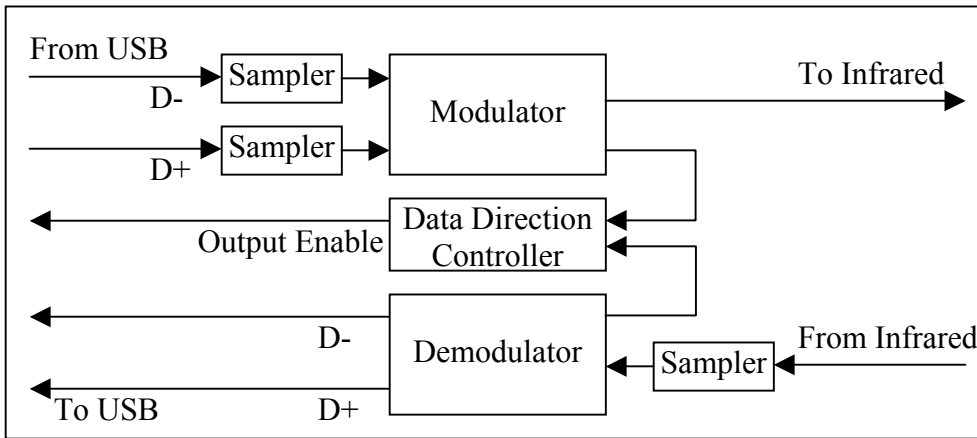


**Figure 10 Dynamic Device Detection Scheme**

Using this dynamic device detection scheme from the USB specification, we are able to embed the device connectivity information in the data lines and send the information across the wireless link.

## 4 USB-IR Bridge

The USB-IR bridge module serves two important functions. It bridges between the USB and infrared module and manages the data flow of the entire system. There is one bridge on each side of the wireless link. The module consists of four main components: samplers, modulator, demodulator and data direction controller. The configuration is show in Figure 11. All the components of the USB-IR bridge are finite state machines programmed using Altera EPM7128S programmable logic device.



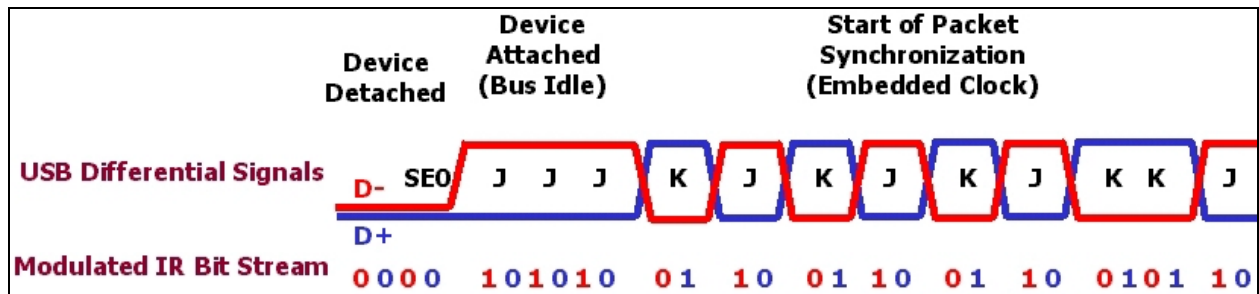
**Figure 11 USB-IR Bridge**

### 4.1 Modulator

The main functions of the modulator are to modulate the USB differential signals into a serial bit stream for the infrared module and to notify the data direction controller when valid data arrive from the differential data lines.

### 4.1.1 Modulator Functions

The modulator uses a simple pulse positioning modulation. The modulator positions the D- signal in the first half of the bit time and positions the D+ signal in the second half. Figure 12 illustrates such a modulation scheme. A J-state is represented differentially by a high D- line and a low D+ line, so a J-state would be modulated into the bit stream '10.' Similarly, a K-state would be modulated into '01,' and a single-ended zero would be modulated into '00.' The bit sequence '11' is reserved for error detection purposes. In this modulation scheme, a bit sequence with more than two consecutive 1's is deemed an erroneous packet.



**Figure 12 Pulse Positioning Modulation**

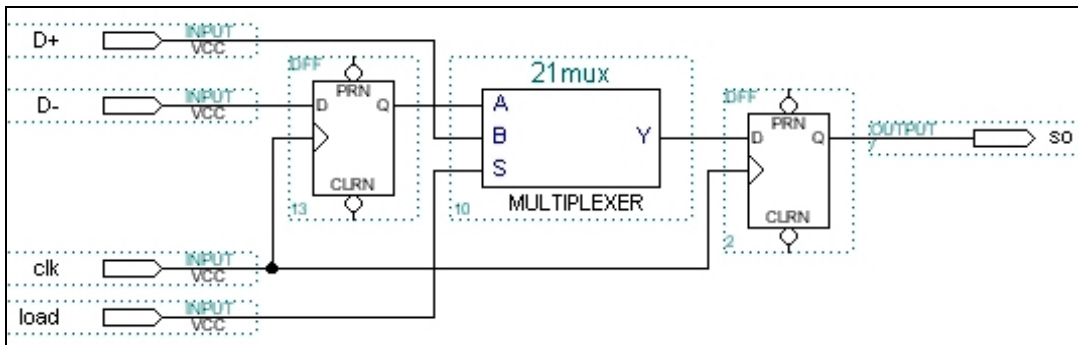
The modulation requires the output data to have twice the USB data rate, which is 3Mbps. The system clock is, for this reason, set at twice the USB data rate. The USB bus clock is embedded in the start of packet synchronization field. For the modulated signal on the other hand, the bus clock is preserved in the idle J-state. A bit stream of consecutive idle J-states is modulated into oscillatory bit stream ('10 10 10...'). This is also illustrated in Figure 12.

The modulator communicates with the data direction controller with an output enable request signal. When the modulator receives valid data from the differential bus lines, it

pulls the request signal high. When the differential bus line becomes idle, it switches the request signal to low.

### 4.1.2 Modulator Implementation

The modulator is implemented using a finite state machine as a sequence recognizer. Its basic function can be characterized by the simple schematic, consisting two shift registers and one multiplexer, shown in Figure 13. The control signal “load” oscillates periodically with a frequency of twice the USB data rate. This results in alternate outputs of D- and D+ data lines at the output pin “so”.



**Figure 13 Basic Function of Modulator**

The modulator needs to notify the data direction controller the status of the modulator. When the modulator detects a start of packet synchronization sequence from the differential lines, it recognizes it as a valid data transfer and notifies the controller with the output enable request signal. When the modulator detects an end of packet SE0 followed by an idle bus, it turns off the request. The recognition of the start-of-packet and the idle bus is achieved by using a finite state machine as sequence recognizer, identical to the one utilized for the demodulator. The finite state machine will be introduced in the demodulator section. The VHDL code for the host modulator is included in the appendices.

4.1.3 Modulator Simulation

The simulation of the bridge has desired results. Figure 14 is a simple simulation of an abbreviated start of packet sequence. The inputs are the D-/D+ differential lines, and the outputs are the serial bit stream “s” and the output enable request “oe.” For the input “KJKJKKJ,” the modulator outputs the correct bit stream “01 10 01 10 01 01 10.” The modulator turns on the output enable request when the idle bus switches to K-state, and disable the request immediately after detecting the end of the packet.

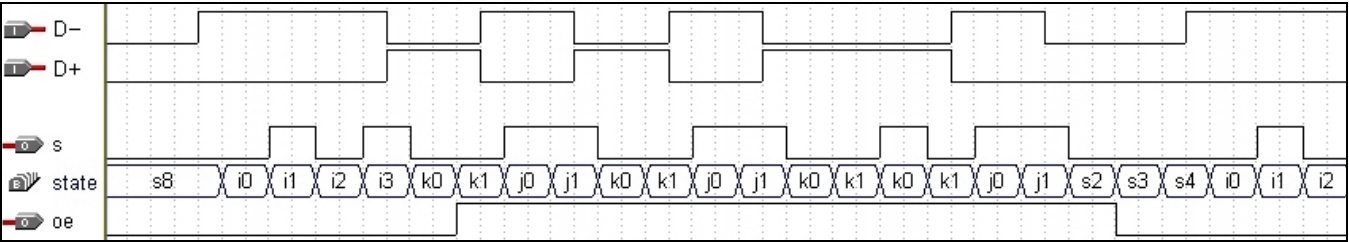


Figure 14 Modulator Simulation

4.2 Demodulator

The main functions of the demodulator are to demodulate the infrared serial bit stream into USB differential signals and to notify the data direction controller on its current status.

4.2.1 Demodulator Functions

The primary function of the demodulator is to convert the modulated infrared signal back into USB differential signals. Table 1 on the right illustrates how the modulated infrared signals would be demodulated.

Modulated Infrared Bit Stream	Demodulated Differential USB State
10	J
01	K
00	SE0

Table 1 Modulation Map

The demodulator needs to inform the data direction controller with two pieces of information: the current status of the demodulator and the connectivity of the device. The demodulator sends an output enable request signal to the data direction controller when valid data arrives from the infrared receiver, resulting in an active demodulator. It also needs to be sensitive of the connectivity of the device at all time. It informs the data direction controller with a disconnection notice immediately after the device is disconnected from the port or the infrared link is blocked.

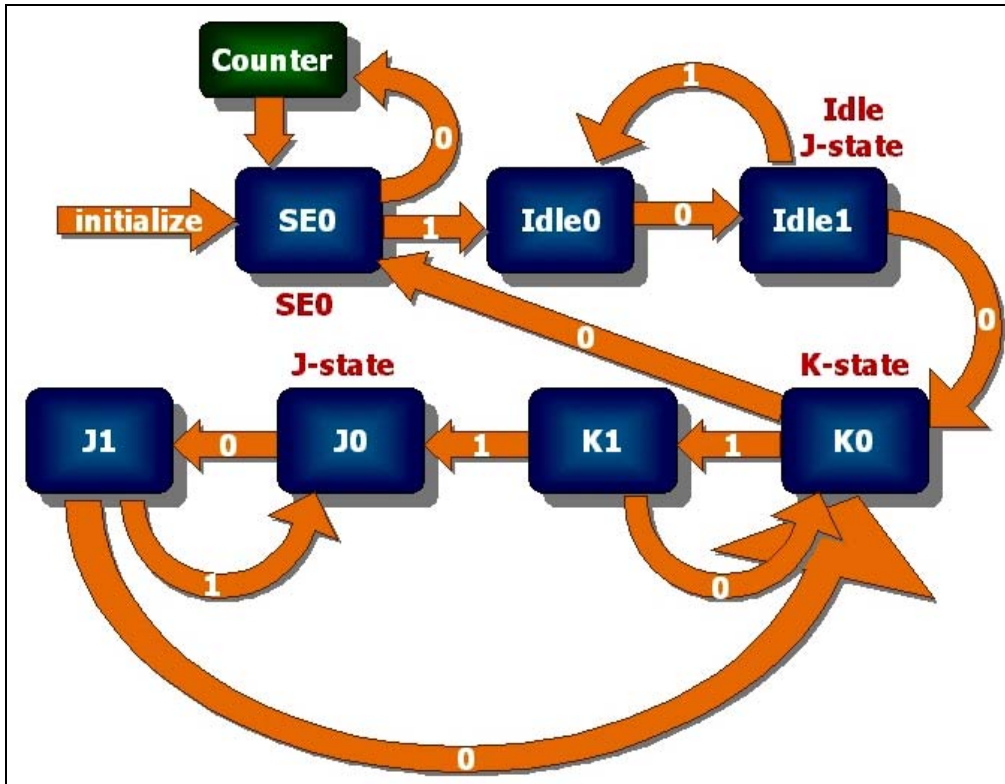
The demodulator has the capability to discern an *idle J-state* from a *data J-state*. This capability is a requirement. The output of a particular data flow needs to be disabled when an idle bus is detected and enabled when a data input is detected.

The demodulator also has the capability to discern an idle bus and a blocked infrared link. This is also an important feature. As a design choice, we will treat a blocked infrared link as a disconnected device, which is represented by single-ended zeros and requires re-enumeration. Using this scheme, a blocked infrared link is effectively differentiated from an idle bus.

#### **4.2.2 Demodulator Implementation**

The demodulator is again implemented using a finite state machine. The full VHDL code is included in the appendices. Figure 15 is the state diagram of the machine. The input to the state machine is the serial bit stream from the infrared receiver. There is an “SE0”

state for the end of packet or a disconnection. There are two idle states, “Idle0” and “Idle1,” and four data states, “K0”, “K1”, “J0” and “J1.” The state machine runs on a clock of twice the USB data rate.



**Figure 15 Demodulator State Diagram**

The demodulator’s primary function is to recognize the bit stream and convert it to differential bus signals. The machine initializes in the “SE0” state. It will remain at this state until a bit 1 is encountered. Meanwhile, it counts the number of bit 0’s. On the eighth count (four bit times), it sends out a disconnection notice to the data direction controller. The output enable request is off at this state.

When a bit 1 is encountered, the machine advances to state “Idle0” and prepares to see an oscillatory idle J-state. The machine then oscillates between state “Idle0” and “Idle1” during idle. The output enable request remains off for the idle bus.

The idle bus is then activated with a synchronization field, which begins with a K-state. The states “K0” and “K1” follow the detection of the K-state bit sequence “01.” The output enable request is turned on immediately at this point and the differential K-state is driven onto the bus.

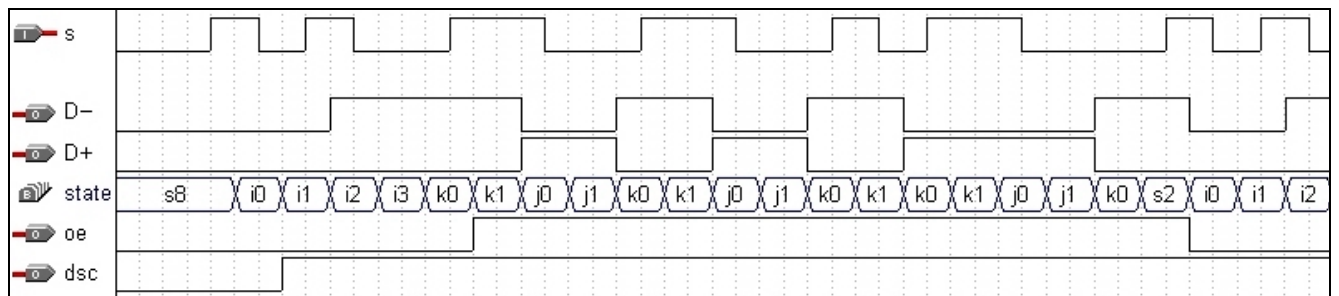
The data payload of a packet consists of J-states and K-states. The state machine thus stays within the four states, throughout the duration of the data payload. The output enable request remains on within the four data payload states.

During any of the data states and the idle states, the state machine is brought back to the “SE0” if a single-ended zero is detected. The counter then counts consecutive single-ended zeros and determines if it is an end of a packet or a disconnected device. For an end of a packet, the bus returns to idle after two bit times. The sequence is repeated.

It is not possible to discern an idle J-state and a data J-state simply by comparing their lengths in bit time. It is allowed in the USB specification to have six consecutive J-state data stream <sup>[13]</sup>. Using data length comparison, the delay will be increased by six bit times! On the other hand, in our demodulation design, an idle J-state and a data J-state is clearly differentiated. An idle J-state is always immediately preceded by single-ended zeros. A data J-state is always buried in a data payload, which is preceded by a start of packet synchronization field. This demodulation design contributes little to the overall logic delay time.

### 4.2.3 Demodulator Simulation

Figure 16 demonstrates the simulation of the VHDL code. Again, we use an abbreviated synchronization bit stream to illustrate the proper workings of the demodulator. The input is the modulated serial bit stream “s.” The outputs are the differential signals D- and D+, the output enable request “oe,” and the disconnection notice “dsc,” which is active low.



**Figure 16 Demodulator Simulation**

At the beginning of the sequence, the bus goes from disconnected to idle. This sequence is represented by the modulated bit stream “00 10 10.” The differential output signals respond with an idle J-state. At this point, the disconnection notice “dsc” (active low) is disabled.

Following the idle J-states, there is an abbreviated synchronization field, represented by the bit stream “01 10 01 10 01 01 10.” The differential D- and D+ outputs the sequence “KJKJKKJ” in respond. Before driving the first K-state onto the line, the demodulator switches on the output enable request, such that the data can be successfully transmitted. The output enable request is switched off when an end of packet single-ended zero is received.

### **4.3 Data Direction Controller**

Data direction controller manages the direction of the data flow on the bus. It controls the data direction by disabling the modulator or the demodulator. The purpose of the controller is to reinforce the half-duplex attribute of the USB differential lines at all time. Under any circumstances, the bus is driven at most on one side at a time.

#### **4.3.1 Controller Design Plan**

The controller acts as an arbitrator in the following situations:

1. The device is disconnected or the infrared link is blocked.
2. Both ends are actively driving the bus at the same time.
3. Both ends are idle at the same time.

The controller is implemented using a finite state machine. The full VHDL code is included in the appendices. Figure 17 shows the state diagram of the controller state machine. The controller requires three inputs: the output enable requests from the modulator and from the demodulator, and the device disconnection notice from the demodulator. It outputs a single data direction control signal DDC. The control signal DDC controls the data direction in the following way:

- When  $DDC = 0$ , the modulator is disabled, and the demodulator is enabled.
- When  $DDC = 1$ , the modulator is enabled, and the demodulator is disabled.

It is important to note that, because only one control signal is in use, the modulator and the demodulator are never enabled at the same time under any circumstances.

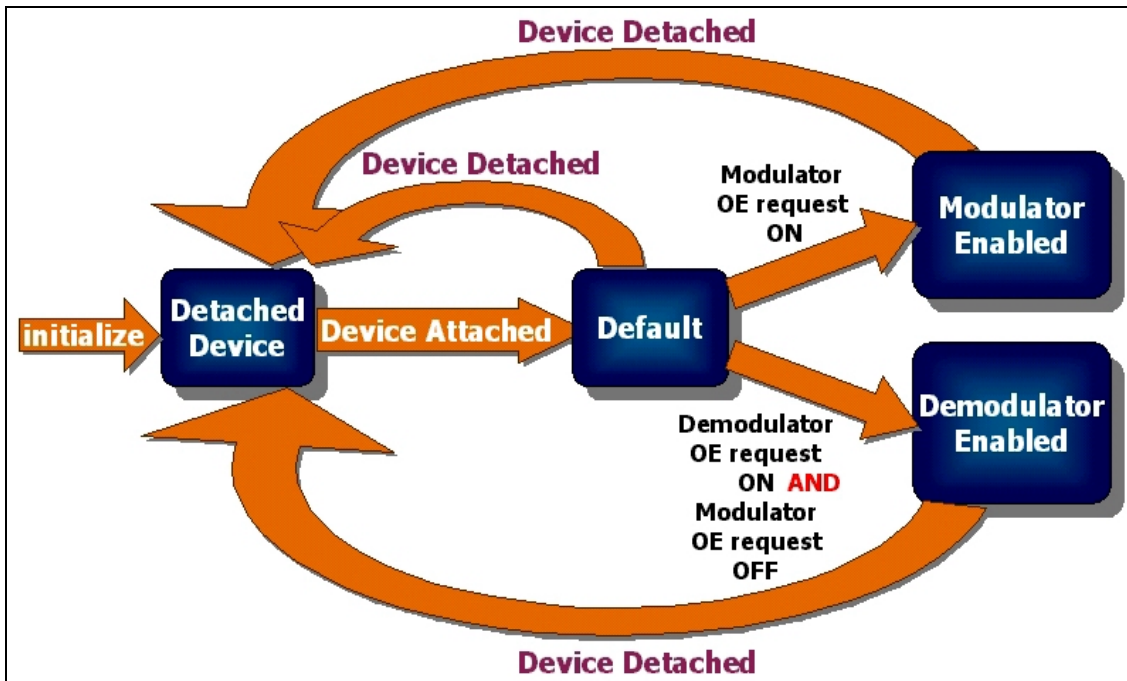


Figure 17 Data Flow Controller State Diagram

#### 4.3.2 Controller State Machine

The machine initializes at the “Detached Device” state. It advances to the “Default” state, when the disconnection notice is disabled.

At the “Default” state, the signal DDC is pulled high to enable the modulator. The controller then checks the output enable requests from the modulator and the demodulator and arbitrates in the following way:

- An output enable request from the modulator is granted an immediate advancement to the “Modulator Enabled” state, and will remain in this state until the modulator request has been switched off.
- An output enable request from the demodulator requires a switched off modulator request to advance to the “Demodulator Enabled” state, and will remain in this state until the demodulator request has been switched off.

The design choice for this discrimination is explained as follows. The modulator sends out its request based on the data from the differential bus lines, while the demodulator sends out its request based on the data from the wireless infrared input. The USB bus data is usually a lot more reliable than the wireless infrared data, hence the discriminatory arbitration scheme.

In the “Default,” “Modulator Enabled” and “Demodulator Enabled” states, the machine returns to the “Detached Device” state whenever a disconnection notice is received from the demodulator. The controller, at this point, needs to set up the bridge, such that the host computer can receive the single-ended zeros to detect the disconnection. There exists a subtle difference between the controllers on the host side and the controller on the device side for this reason. They differ in the following way.

- On the host side,  $DDC = 0$ , such that it is receiving input from the blocked infrared port.

- On the device side, DDC=1, such that it is receiving input from the disconnected differential line.

In both cases, the host is able to receive consecutive SE0's and recognize the disconnection.

## **4.4 Sampler**

The main function of the sampler is to remove undesired glitches from the incoming input signals.

The signal from the infrared receiver goes through a comparator. The comparator outputs the only level high and level low signals, but does not remove glitches from noise. The samplers are incorporated in the USB-IR bridge to resolve this problem.

The sampling clock runs at 16 times the USB data rate (24MHz). Each sampler has a counter, which counts consecutive logic 1's or logic 0's. A signal lasting shorter than half a bit time is considered a glitch and ignored. The counter is reset after removing the glitch. A signal lasting longer than half a bit time is considered a legitimate data and is kept as an input.

The counter is implemented using state machines. Each advancing state represents an increment on a count. The full VHDL code of the counters is included in the appendices.

The input from the USB bus lines have no glitches, but are still fitted samplers, as shown in Figure 11. This arrangement is designed for low speed device only. For low speed transactions, the host sends a special preamble packet at high speed before each packet to warn the downstream hubs of the transaction in low speed <sup>[14]</sup>. These preamble packets need not to be understood by a low speed device <sup>[15]</sup>; therefore, we treat them as noise as well.

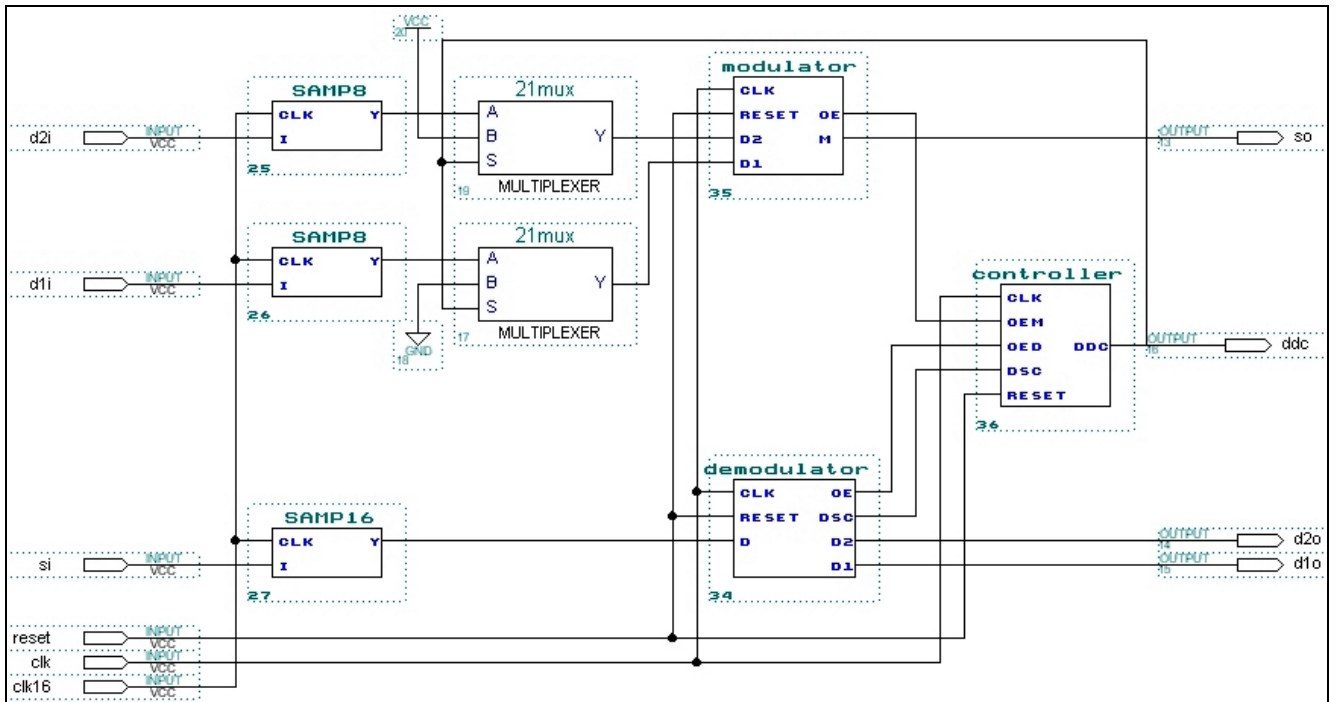
## **4.5 Overall Bridge Circuit**

The overall bridge circuit has components in both transmit and receive direction with a controller managing the data flow direction.

### **4.5.1 Control of Data Flow Direction**

Disabling the modulator or the demodulator achieves the control of data flow directions. The control signal disables them differently. The overall bridge schematic, shown in Figure 18, illustrates the role of the control signal DDC.

The modulator is disabled when  $DDC = 0$ . From Figure 18, DDC is connected as the control signal for two multiplexers preceding the input to the modulator. When the controller wishes to disable the modulator and pulls DDC low, the multiplexers output from input “B.” The two “B” inputs are connected to a VCC and a ground, forming an idle J-state and effectively disable the modulator.



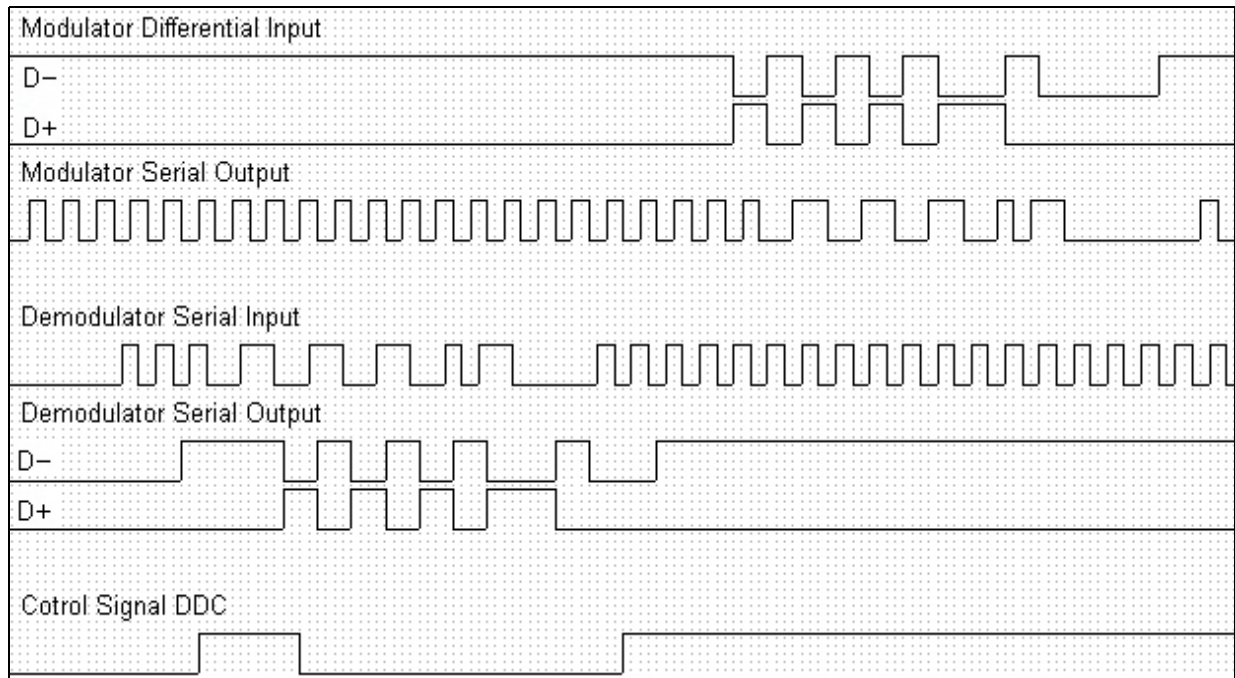
**Figure 18 Overall Bridge Schematic**

The demodulator is disabled when DDC = 1. The DDC output in Figure 18 is fed to the output enable (OE#) pin on the USB transceiver of the USB module. The OE# pin is active low; thus, an input high disables the transceiver driver. The USB transceiver effectively blocks off and disables the demodulator in this situation.

#### 4.5.2 Overall Bridge Simulation

Figure 19 is the simulation of the overall system. The simulation is done using standard USB synchronization fields as inputs. At the beginning of the sequence, the device is connected to the port and the control signal DDC raises to its default state. The Demodulator encounters an input first. The control signal DDC is then pulled low to allow the demodulator to drive the differential bus. The demodulator correctly converts

the serial input into differential signals. When the demodulator encounters the end of packet single-end zeros, DDC switches back to its default value for a possible data input to the modulator.



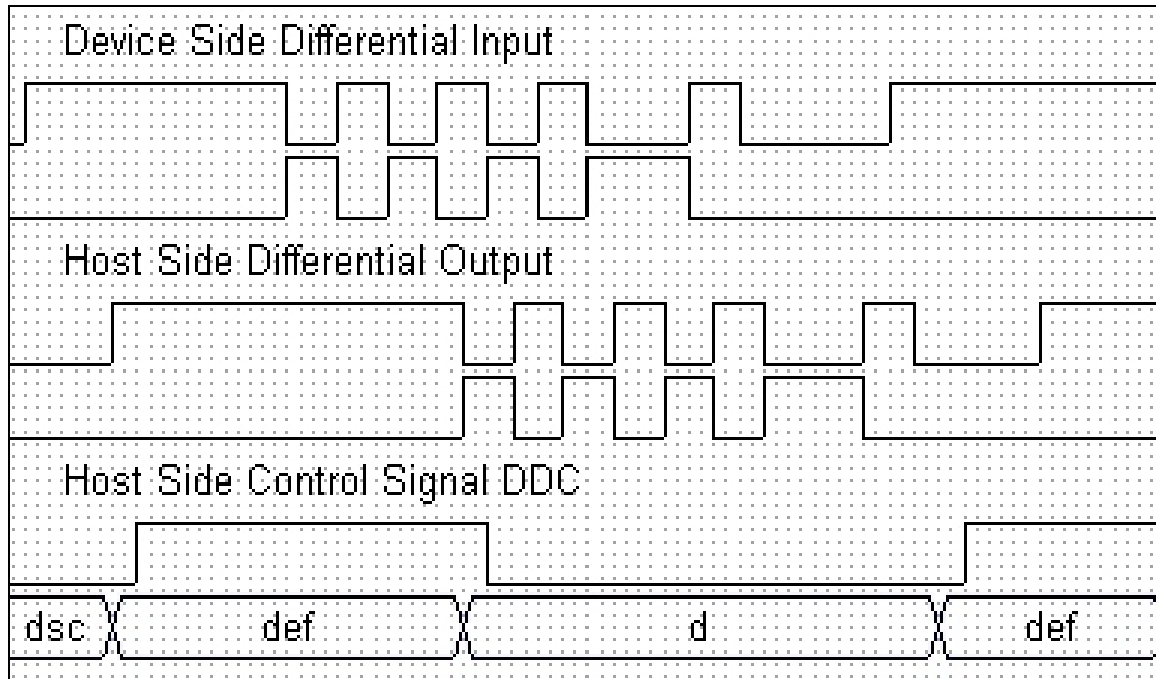
**Figure 19 Bridge Simulation**

### 4.5.3 USB-IR Bridge Performance

To test the performance of the bridge, a direct link between the device side and the host side is simulated.

Figure 20 illustrates the result of such a test. This simulation combines the USB-IR bridges on both side of the wireless link and replaces the wireless link with a logical connection inside the programmable logic device.

The test shows the direct link works properly. The test input gets modulated on the device side and demodulated on the host side. The output on the host side echoes the input on the device, which shows the correct working of the bridges.



**Figure 20 Simulation of a Direct Connection**

This simulation demonstrates that the two bridges combined have a total logical delay of 3.5 bit times, as shown in

Figure 20. The propagation delays of the USB transceivers and the programmable logic device combined are on the order of 100ns. We can conclude that, without the infrared link, the interface has a delay time of four bit times. This delay time is better than four times the required minimum delay time mentioned in Chapter 2.

This simulation also demonstrates another advantage of our bridge design. The entire USB-IR bridges, on both the host side and the device side, use less than 80% of the logic

cells on the Altera EPM7128S programmable logic device. Each bridge contains, not only the main modulation components, but also samplers and multiplexers. In the logic size perspective, this design is far more superior to the bridge of the other groups. There is also a possibility in the reduction of the physical size of the PCB layout as well, since each bridge uses less than 40% of the logic cells.

## **5 Infrared Module**

This section covers the research, design and testing on the infrared transceiver module accomplished during the process of design project.

### ***5.1 Technical Issues on Infrared Transceiver Design***

The merit of optical communication is its capability and potential of extremely high transmission rate. Typically, the wavelength for near infrared communication is from 850nm to 900nm in free space, which correspond to a frequency from 333THz to 353THz. The high frequency implies the ultimate bandwidth it's capable of.

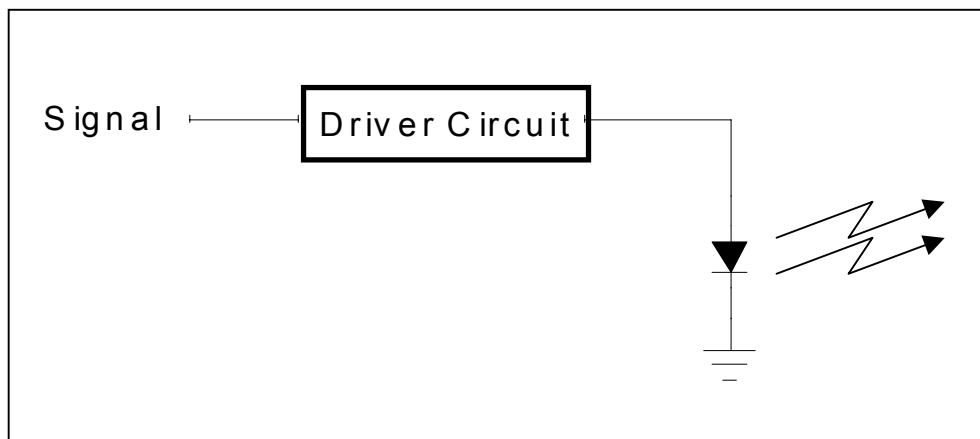
However, constraints on open space connection and transient response of photoelectronics, the data transmission speed is restricted. The nature of wireless connection introduces multiple reflections of transmitted data beams, which result in dispersion in receiver end. Yet, the transmission power of infrared LED is comparably small such that the dispersion effects will be weighed down by issues on the speed of photoelectronics.

The fundamental concerns on designing a high-speed infrared transmitter and receiver will be the limitation of photoelectronic devices, specifically on the transient response such as rise time, fall time of LED, MOSFET, phototransistor, photodiode, operational amplifier and comparator.

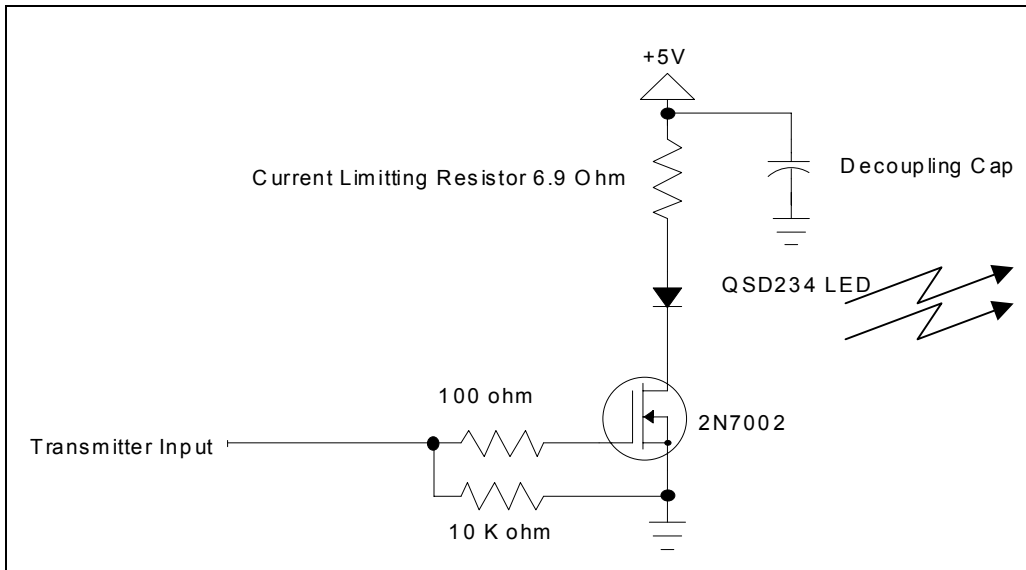
In order to transmit modulated USB signal from USB-IR bridge, the designed infrared transceiver is required to have minimum speed of 3.0 MHz.

## 5.2 Transmitter Design

The main function of transmitter is to convert electrical signal into optical signal. An infrared Light Emitting Diode, the main component of transmitter, emits photons when stimulated by current. In order to provide current flowing through LED in respond to voltage input, a driver circuit will have to be premeditated (see Figure 21 for the topology of transmitter). The driver circuit is designed to produce a current going through infrared LED when input signal of 1 is being fed into gate of MOSFET. An n-channel MOSFET will be apt to drive current in respond to voltage input. The circuit design of transmitter is shown in Figure 22<sup>[16]</sup>. When the signal is 1 (which corresponding to  $V_g=5$  volts), a drain current of MOSFET will go through diode and produce infrared emission.



**Figure 21 Topology of Infrared Transmitter**



**Figure 22 Infrared Transmitter Circuit**

### 5.2.1 Selection of Light Emitting Diode

Some of the important features of infrared LED are its peak emission wavelength, emission angle, radiant intensity and rise time/fall time. The peak emission wavelength will have to match with the peak sensitivity wavelength of infrared diode or transistor to ultimate the received signal intensity. Radiant intensity states the amount of power the amount of power LED is emitting per unit current. Larger power corresponds to larger signal intensity and higher fidelity at receiver end. A wider emission angle will provide a better dynamic range for the wireless link. However, the radiant intensity would be smaller for large emission angle. Rise time and fall time are the main issues for speed consideration. In order to achieve a speed of 3Mbps, the maximum rise time and fall time shall not exceed 40ns<sup>[17]</sup>. Several types of Infrared LED are compared in Table 2. The LED's, which meet the speed requirement are HSDL-4220 and HSDL-4230.

However, due to the availability of LED samples, the testing transmitter circuits are built with QSD234 from Fairchild. Since the rise time/fall time of LED (1000ns) become quite comparable to bit time (333ns), several adjustments are required in receiver design discussed in next chapter.

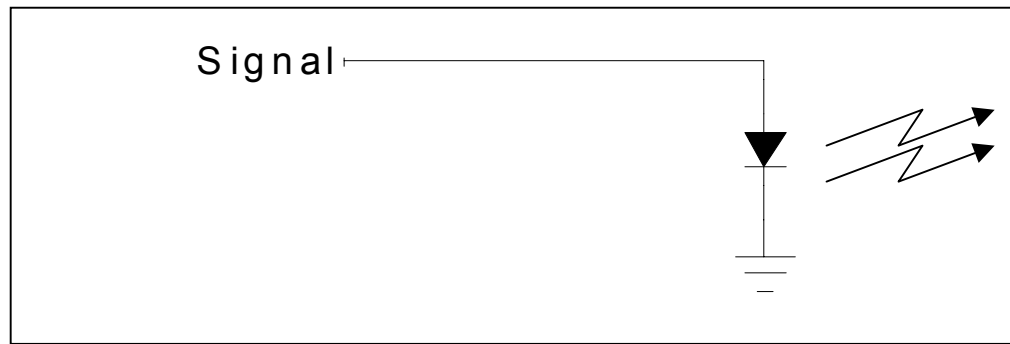
Device	Material	Peak Emission Wavelength	Emission Angle	Radiant Intensity	Rise/Fall Time	Availability
QEC234	GaAs	940 nm	40 Deg.	10 mW/sr	1000ns/1000ns	Yes
QEC113	GaAs	940 nm	24 Deg	14 mW/sr	1000ns/1000ns	Yes
HSDL-4220	AlGaAs	875 nm	30 Deg.	22 mW/sr	40ns/40ns	No
HSDL-4230	AlGaAs	875 nm	17 Deg.	39 mW/sr	40ns/40ns	No

**Table 2 Comparison of Available Infrared LED**

## 5.2.2 Need for Driver Circuit

LED emits photocurrent by spontaneous emission when it is forward biased <sup>[18]</sup>.

However, directly feeding LED with signal voltage has several drawbacks (Figure 23).



**Figure 23 Directly feeding LED with signal**

Firstly, loading a LED (which can be considered as a series of resistor and ideal diode) will drop the signal voltage level which could result in some unwanted loading effects. Moreover, as battery degrades, the voltage power level will drop, which could results in lower driving current. A driver circuit is needed to compensate these problems. An ideal

driver circuit will have to provide a constant driving current regardless of voltage supply voltage and minimize the loading effect.

### 5.2.3 Design of Driver Circuit

To maximize the intensity of output optical signal, the driving current is proposed to be the maximum current rating for LED, i.e. 100mA. The output power intensity of LED is 27 mW/Sr. It also meets the IEEE Eye Safety Standard, which is under 500 mW/Sr

Both MOSFET and BJT can provide constant driver current. MOSFET is preferred in our design because it has minimum loading effect. It provide constant current flow through LED in response to only gate voltage as long as the MOSFET is operating in saturation region. Ideally, there is no current going into gate of MOSFET which means the signal is not loaded with LED.

	<b>2N7000</b>	<b>2N7002</b>
Manufacturer	Fairchild	Fairchild
Turn On Time	10ns	20ns
Turn Off Time	10ns	20ns
Threshold V	2.1V	2.1V

**Table 3 n-channel MOSFET Specifications**

Two types of MOSFET's listed in Table 3 are suitable for our design. The signal input is connected to gate of MOSFET with a resistor with low resistance. Signal input is also connect to ground with a resistor which drains current from signal input. The resistor chosen here is desirably large in order to minimize power dissipation ( $P = V^2/R$  and V is fixed). LED is forward biased at drain of MOSFET with a current limiting resistor. The

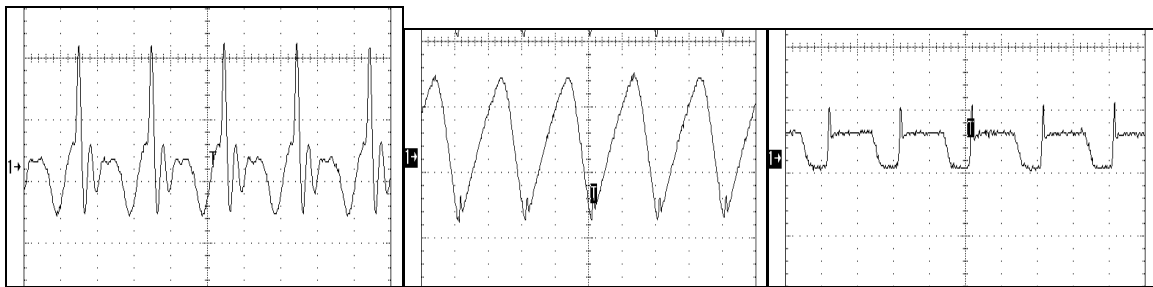
current limiting resistor is obtained experimentally to meet the proposed driving current, i.e. 100mA (Table 4).

Current Limiting Resistance	Voltage Drop across Resistor (mV)		Driving Current (mA)
	V <sub>in</sub> = 1	V <sub>in</sub> = 0	
3.9 Ohm	760	0	194.8
6.8 Ohm	752	0	110.6

**Table 4 Experiment result for selection of current limiting resistance**

### 5.2.4 Decoupling Capacitor

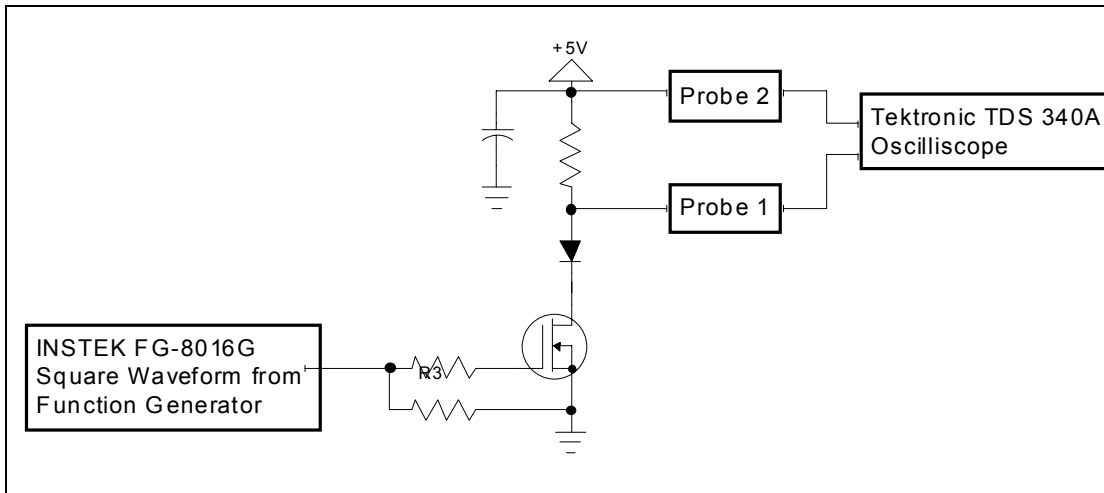
It is observed that when testing transmitter design on breadboard, there is severe interference from function generator on V<sub>cc</sub> (Figure 24 (a)). V<sub>cc</sub> node oscillates dramatically with frequency same as input signal. This can be improved by imposing a decoupling capacitor between V<sub>cc</sub> and ground. Different value of capacitor is tested and the larger the capacitor, the better the decoupling effect(Figure 24(b), (c)). Decoupling capacitor is chosen to be 10  $\mu$ F with sufficient stability of V<sub>cc</sub>.



**Figure 24 V<sub>cc</sub> Oscillation with (a) no decoupling capacitor (V<sub>pp</sub> = 5.6V) (b) Cd=0.1 $\mu$ F (V<sub>pp</sub> = 228mV) (c) Cd=10 $\mu$ F (V<sub>pp</sub> = 38mV)**

### 5.2.5 Driver Circuit Testing

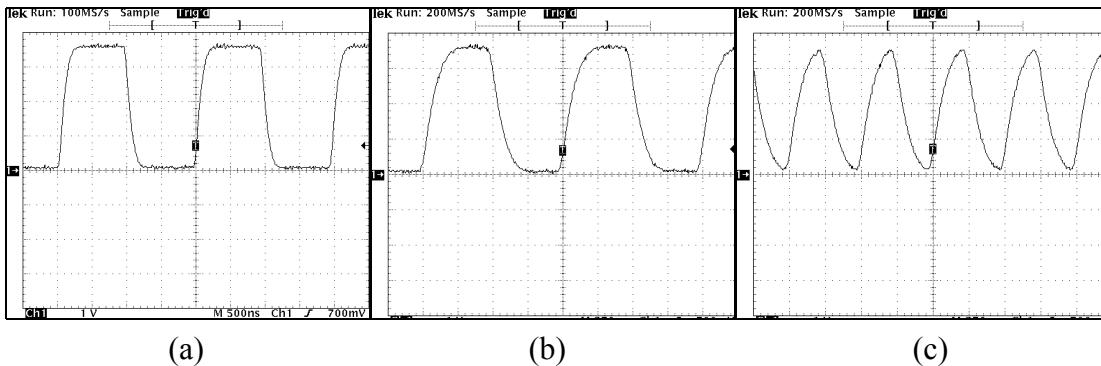
The voltage and current characteristic are tested with the following experimental setup:



**Figure 25 Experiment setup for measuring driving current**

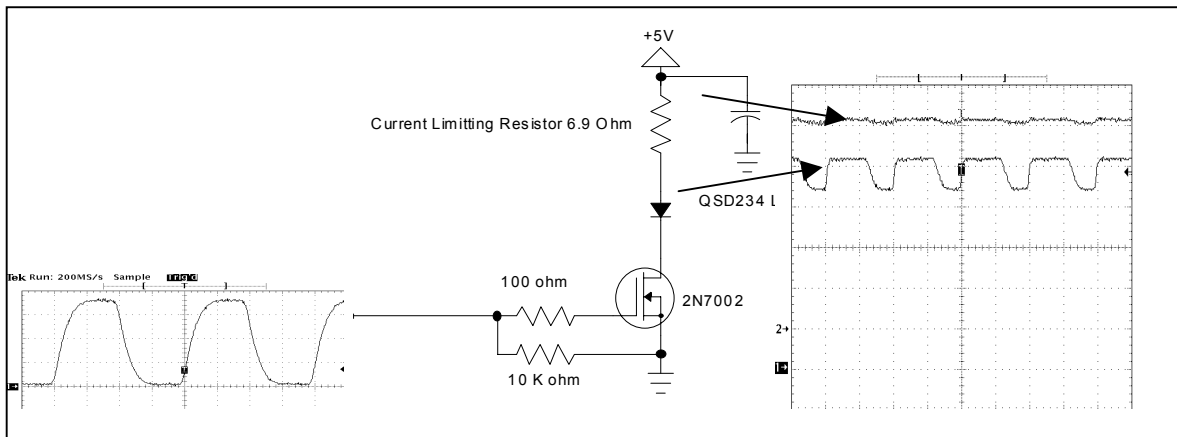
The function generator is working well under frequency of 1MHz(Figure 26(a) and (b)).

The waveform looks distorted when frequency is over 2MHz (Figure 26(c)). As a result, the testing is done under 1Mhz with the provided function generator.



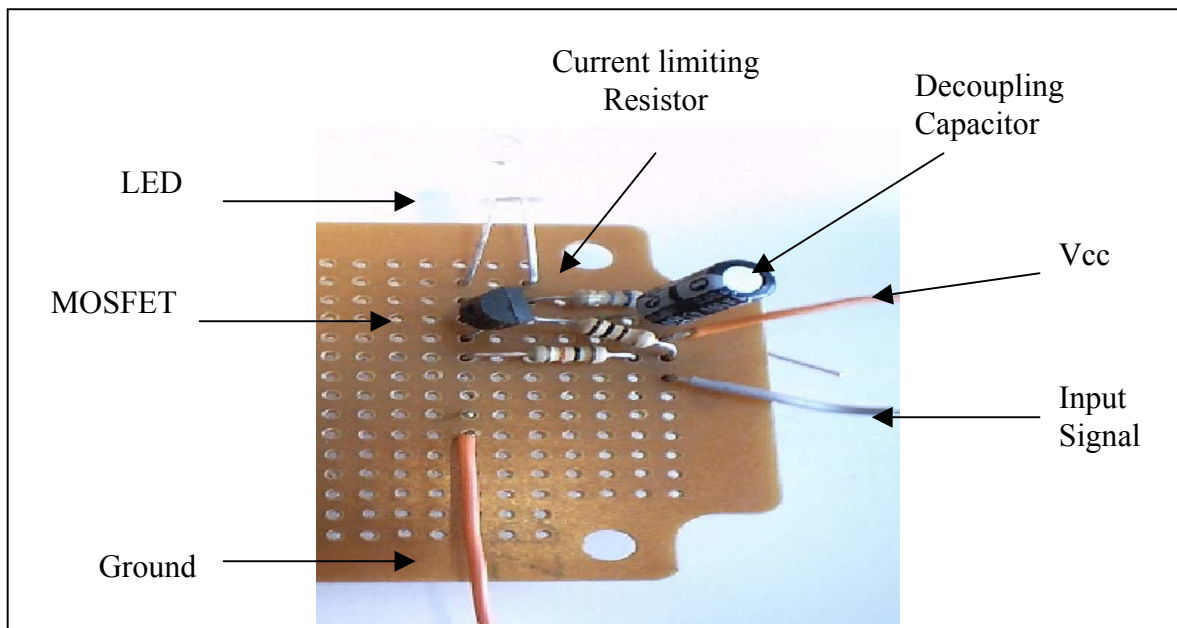
**Figure 26 Function Generator Output at frequencies of (a) 500KHz (b) 1MHz (c) 2MHz**

Two probes from oscilloscope are connected to the two ends of current limiting resistor to determine driving current. Figure 27 shows the output waveform on oscilloscope.



**Figure 27 Output Waveform**

The measured current through resistor is 111.3mA. The transmitter circuit is build and tested on vector board (Figure 28), which would minimize the inductance effect on breadboard.



**Figure 28 Transmitter circuit built on vector board**

### 5.3 Infrared Receiver Circuit Design

The design of Infrared Receiver requires more deliberations. Besides desired infrared data signal, photodiode or phototransistor would also receive other light emissions from sunlight, light bulbs, or fluorescent lamps, which constitute noise sources. The receiver will have filter out these unwanted signals. Owing to the wireless link nature, the signal intensity is a variable of distance from transmitter to receiver. The receiver will also be able to pick up signal of dynamic intensity and translate them into output signals of fixed intensities. Several inherent noises from circuits also possibly exist in our circuit design, for example dark current of photodiode and shot noise from photodiode when DC-biased. These noises will also have to be filtered out with appropriate filter circuits.

#### 5.3.1 Selection of Photo Detector:

The common two types of photo detectors are photodiodes and phototransistors.

Selected photodiode and phototransistor are listed in Table 5.

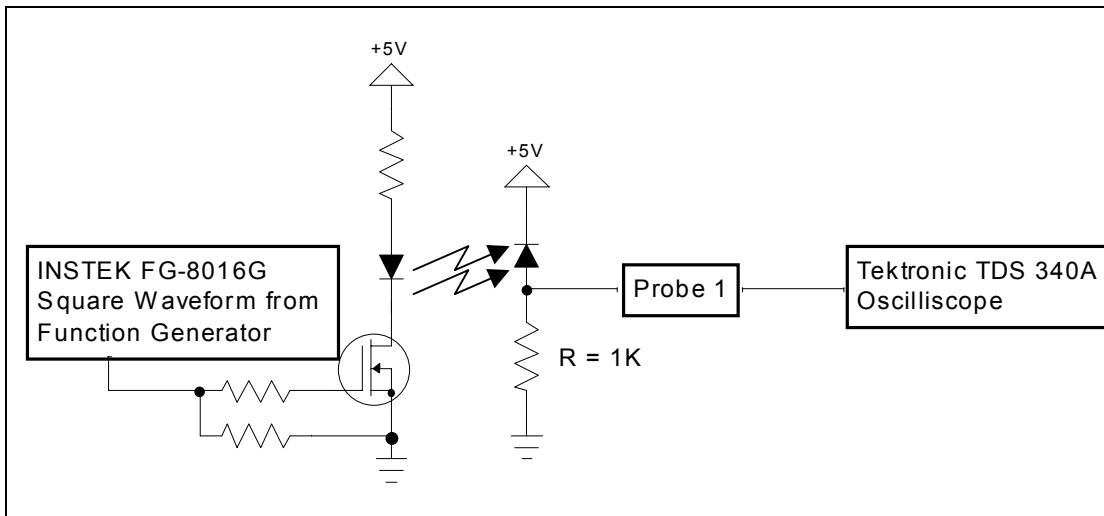
Manufacturer	Fairchild	Fairchild	Fairchild	Fairchild
Type	Plastic Silicon Photodiode	Sidelook Photodiode	Silicon Phototransistor	Silicon Phototransistor
Model	QSE773	QSE973	QSD124	QSD128
Peak Sensitivity	920 nm	930 nm	880 nm	880 nm
Reception Angle	+/- 60 degree	+/- 45 degree	+/- 12 degree	+/- 12 degree
Photo Current	30 uA/mW/cm <sup>2</sup>	30 uA/mW/cm <sup>2</sup>	12 mA/mW/cm <sup>2</sup>	3.2 mA/mW/cm <sup>2</sup>
Rise time	50 ns	50 ns	7us	7us
Fall Time	50 ns	50 ns	7us	7us
Dark Current	30 nA	30 nA	NA	NA
Availability	YES	YES	YES	YES

**Table 5 Comparisons of Available Photodiodes and Phototransistors**

From Table 5, phototransistor has greater sensitivity than photodiode. Nevertheless, the superior transient response of photodiode indicates higher speed capability. Therefore, photodiodes will be used in our design.

### 5.3.2 Photocurrent with varying distance

It is necessary to know the magnitude of photocurrent in response to transmitter with different distances. The test for measuring photocurrent with varying distance is done in the following setup:



**Figure 29 Experiment setup for measuring relationship between photo current and distance**

The experiment results are summarized in the Table 6

Distance(cm)	Resistance	Vpp(V)	Vmean(V)	Ipp (uA)	Imean(uA)	Imax (uA)	Imin (uA)
0	1000	0.960	1.235	960.0	1235.0	1715.0	755.0
1	1000	0.240	0.284	240.0	283.7	403.7	163.7
2	1000	0.088	0.098	88.0	98.4	142.4	54.4
4	1000	0.024	0.031	24.8	31.3	43.7	18.9
8	1000	0.010	0.013	10.0	13.4	18.4	8.4
12	1000	0.006	0.010	6.4	10.1	13.3	6.9
0	100	0.160	0.149	1600	1488.0	2288.0	688.0
1	100	0.034	0.028	344	279.3	451.3	107.3
2	100	0.017	0.015	168	154.1	238.1	70.1

**Table 6 Photocurrent measurement with varying distance**

From the output waveform from oscilloscope, it is observed that the signal level doesn't return to zero when no signal is sent at transmitter. This is because the rise/fall time of the system is comparable to one bit time of signal. One contribution to total rise/fall time is the rise/fall time of LED (1000 ns). The reversed biased photodiode had also built up a capacitance of 30pF. Combined with the resistor connected in series, the first stage of receiver also provides a RC time constant of 30ns.

### 5.3.3 Proposed BER and SNR

In a communication system, BER is one the most important measurement on the performance. The BER specified by IrDA is  $10^{-8}$  which mean one bit error in 100 million bits. BER depends primarily by Signal Noise Ratio (SNR). The larger SNR is, the smaller the BER will become. The minimum receiver signal level is estimated as following:

BER is related to Q factor in the following equation<sup>19</sup>:

$$BER = \frac{1}{2} \operatorname{erfc}\left(\frac{Q}{\sqrt{2}}\right) \approx \frac{\exp\left(-\frac{Q^2}{2}\right)}{Q\sqrt{2\pi}}$$

From proposed BER, we can find desired Q factor to be:

$$BER = 10^{-8} \Rightarrow Q = 5.515$$

In ideal case when threshold is half way between maximum received signal and minimum received signal, Q factor is defined by the ratio of peak to peak photocurrent to two times noise current.

$$Q = \frac{I_1 - I_0}{\sigma_1 + \sigma_0} = \frac{I_{pp}}{2\sigma}$$

Noise current is defined as geometric mean of thermal noise current and shot noise current:

$$\sigma = \sigma_0 = \sigma_1 = \sqrt{\sigma_T^2 + \sigma_s^2}$$

Firstly we need to estimate thermal noise current:

$$\sigma_T^2 = \frac{4K_B T}{R_L} F_n \Delta f$$

$$K_B \quad (\text{Plank's Constant}) = 1.38 \times 10^{-23}$$

$$T \quad (\text{Temperature}) = 300 \text{ K}$$

$$F_n \quad (\text{Amplifier Noise Figure}) = 10 \text{ (estimated)}$$

$$R_L \quad (\text{First Stage Resistor Value}) = 1000 \text{ Ohm}$$

$$\Delta f \quad (\text{System Bandwidth}) = 3 \text{ MHz}$$

$$\Rightarrow \sigma_T = 22.3 \text{ nA}$$

Shot noise current is given in the following equation:

$$\sigma_s^2 = 2q(I_p + I_d)\Delta f$$

q (electron charge) =  $1.602 \times 10^{-19}$

$I_p$  ( mean photocurrent ) which is depends on distance

$I_d$  (dark current) = 30 nA

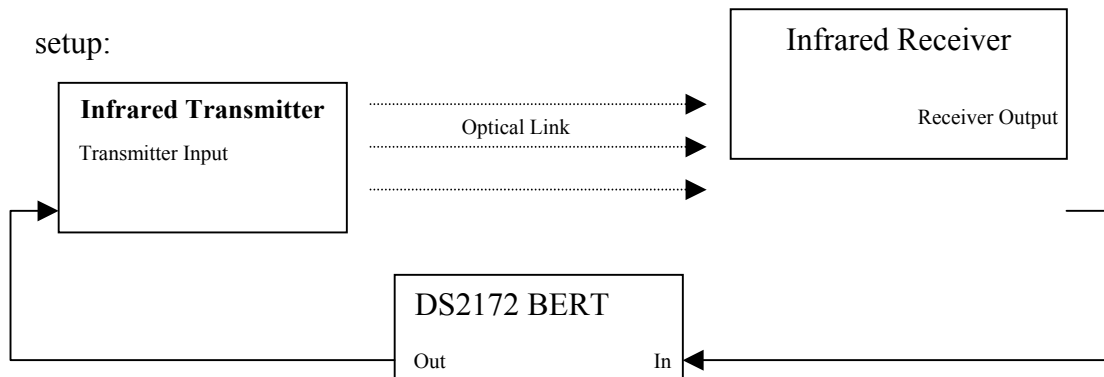
$\Delta f$  (System Bandwidth) = 3MHz

Since mean photocurrent is dependent on distance, the calculation results are summarized in the following table:

Distance (cm)	Mean Photo Current (uA)	Shot Noise Current (nA)	Thermal Noise Current (nA)	Total Noise Current	Required Ipp(uA)	Actual Ipp (uA)
0	1235.00	344.60	22.30	345.32	3.81	960.00
1	283.70	16.50	22.30	27.74	0.31	240.00
2	98.40	9.70	22.30	24.32	0.27	88.00
4	31.30	5.50	22.30	22.97	0.25	24.80
8	13.40	3.50	22.30	22.57	0.25	10.00
12	10.10	3.10	22.30	22.51	0.25	6.40

As a result, photocurrent will have enough intensity to satisfy the proposed BER for all distance we have measured.

However, the BER estimation is subject to many other errors. Many other noise sources like ambient light, additional system noise are hard to be estimated. The best way to obtain BER is to test the overall circuit with a Bit Error Rate Tester in the following setup:

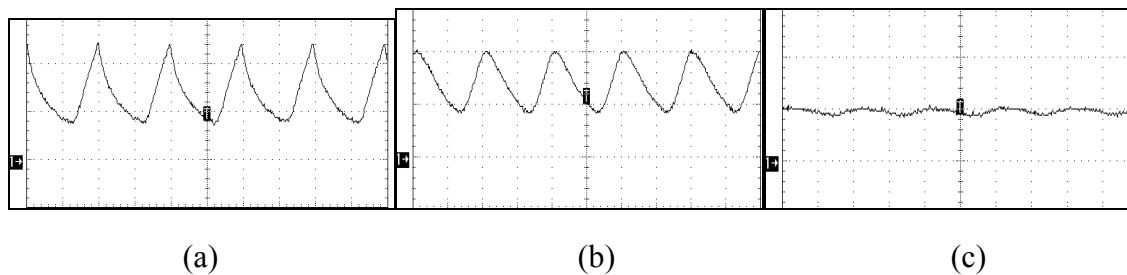


### Figure 30 BER Testing Circuit

The Bit Error Rate Tester shown in the figure is DS2172 manufactured by Dallas Semiconductor.

#### 5.3.4 Selection of Resistance at First Stage

A reversed-biased photodiode act as a capacitor. When it is connected in series with a resistance, it will introduce RC time constant and affect receiver performance. In order to study the influence of RC time constant, different resistors are tested with reversed biased photodiode. The output waveforms of receiver with different resistor value are shown in the Figure 31.



**Figure 31 Output waveform at the first stage of receiver with resistor value of (a) 100 ohm, (b) 1000 Ohm, (c) 10 Kohm**

With smallest resistance (100 ohm), the waveform has the shortest RC time constant by observing that the waveform is rising and falling in a faster rate. Due to limitation on amplifier gain-bandwidth, the voltage level at first stage resistance can not be smaller than 0.1 V. In Table 5, it is shown that the voltage level will need more than 50 times amplification for resistor of 100 ohm. Therefore, resistor of 1Kohm is chosen in our design. When resistor is 10kohm, the rise time and fall time become too large to distinguish (Figure 31(c)).

### **5.3.5 Second Stage: Preamplifier**

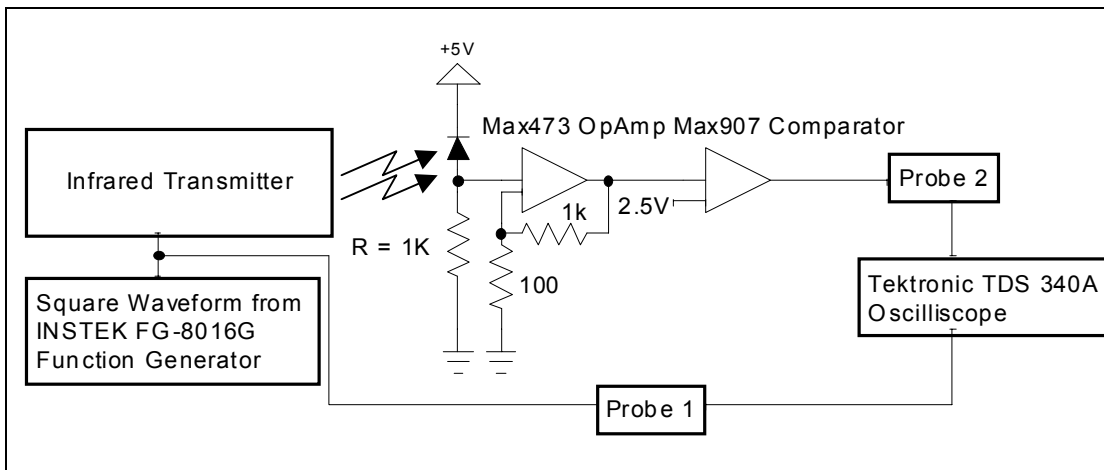
From output voltage in the first stage of receiver in Table 5, the mean voltage ranges from 13mV to 283mV when distance ranges from 1cm to 8cm. The voltage is quite small. Therefore, we need an amplifier to make the signal large enough for further process. A ten times amplifier which amplify signal to a range of 130mV to 2.83V would produce a satisfactory signal amplitude. It is also important to choose an operational amplifier with gain-bandwidth product more than 30 MHz in order to meet the system speed requirement. MAX473 Op Amp from MAXIM Semiconductor is chosen to build the preamplifier in our design.

### **5.3.6 Third Stage: Comparator**

In order to compensate the visible light interference, a physical filter is necessary. All of the four photodiodes or phototransistor are enveloped with daylight filter, which can sift out waves in visible spectrum. Because of dynamic range, intensities of photocurrent and output of pre-amplifier are subject to distance. In order to compensate this problem and produce a steady receiver output, a comparator is employed to produce fixed levels of output signal.

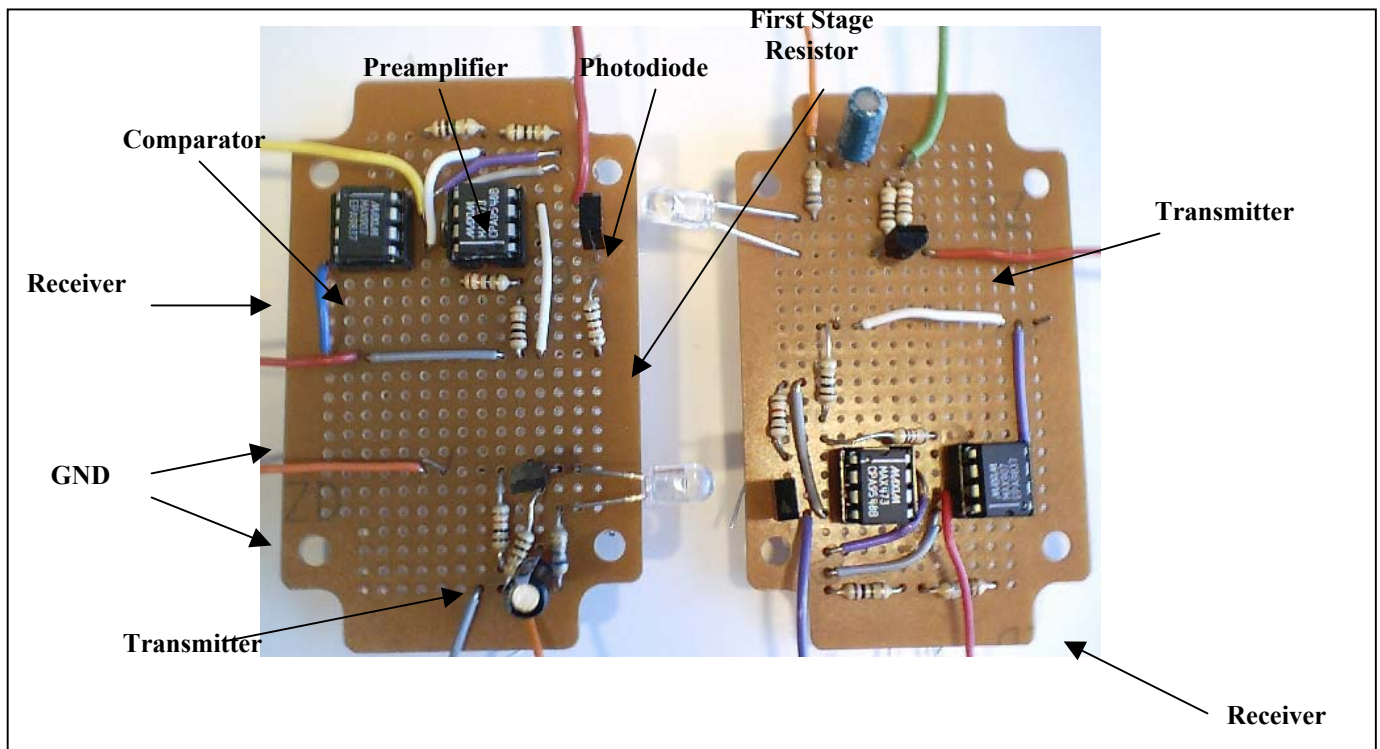
### **5.3.7 Testing circuit with a fixed comparator input**

The overall system is tested with a fixed comparator input of 2.5 V in the following setup:



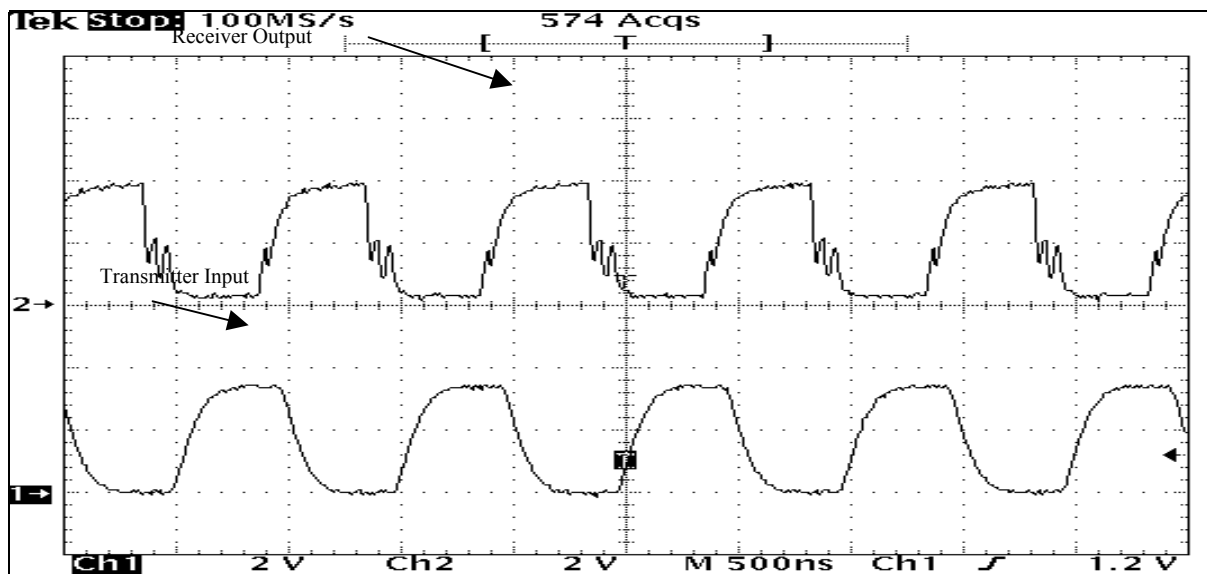
**Figure 32 Experiment Setup for testing receiver circuit with fixed threshold**

Both transmitter and receiver circuits were built and tested on vector board (Figure 33).



**Figure 33 Overall system testing on vector board**

The comparator produces a perfect output waveform (Figure 34) when transmitter and receiver is exactly 5mm apart.

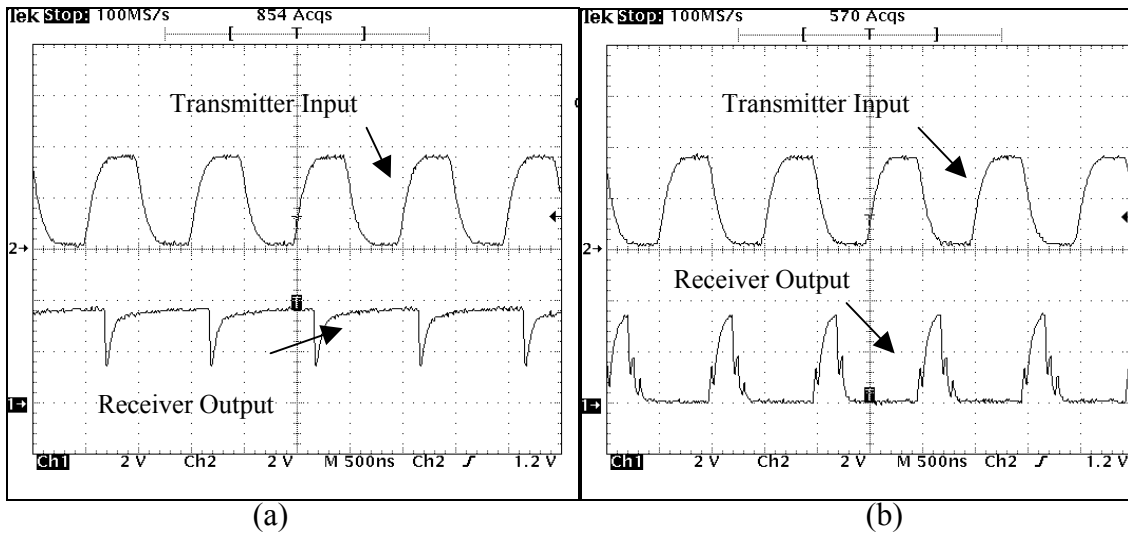


**Figure 34 Input of transmitter and output of receiver with 2.5V threshold level when 5mm apart**

However, the output waveform is very sensitive to distance variation (Figure 35(a), (b)).

When the photodiode is too close to transmitter, the threshold will become lower than the mean value of received signal. Therefore, most of received signal will be larger than threshold and will produce high (1) at comparator output.

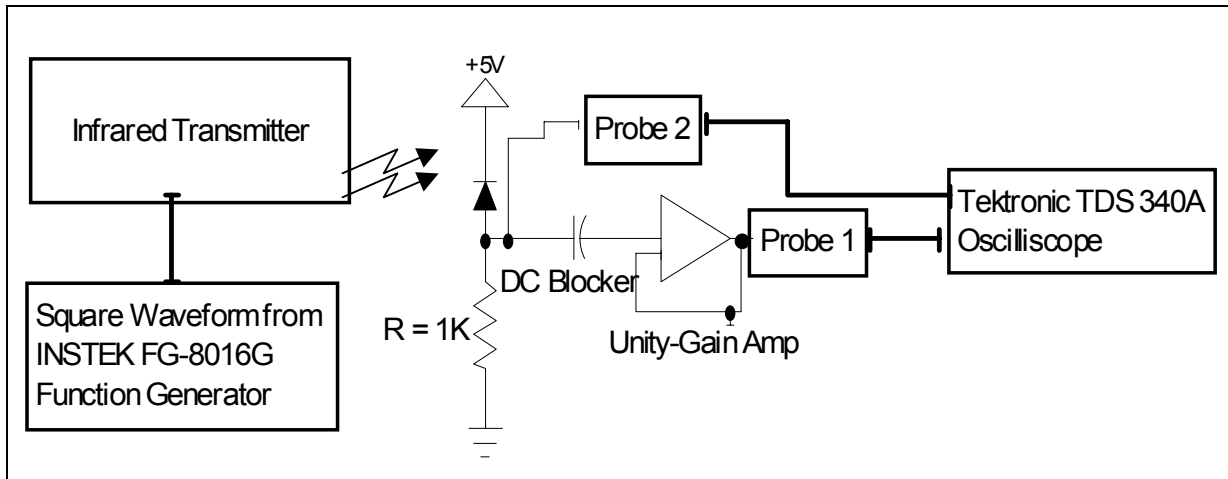
The reason is that the rise/fall time is too large, and a slight change in threshold level would produce a big difference. As a result, it is necessary to provide a precise threshold level for comparator.



**Figure 35 Transmitter input and receiver output when threshold level(2.5V) is (a) smaller than mean value (2mm apart) (b) larger than mean (8mm apart)**

### 5.3.8 DC blocker: Compensation for dynamic threshold

One approach to provide a precise threshold level is to insert a capacitor, which act as a DC blocker. It is place between the first and second stage of receiver. An efficient DC blocker would lower the mean value of received signal close to zero. Therefore, a fixed threshold close to is sufficient to provide accurate threshold level. Several capacitors are tested as DC blocker in the following setup:



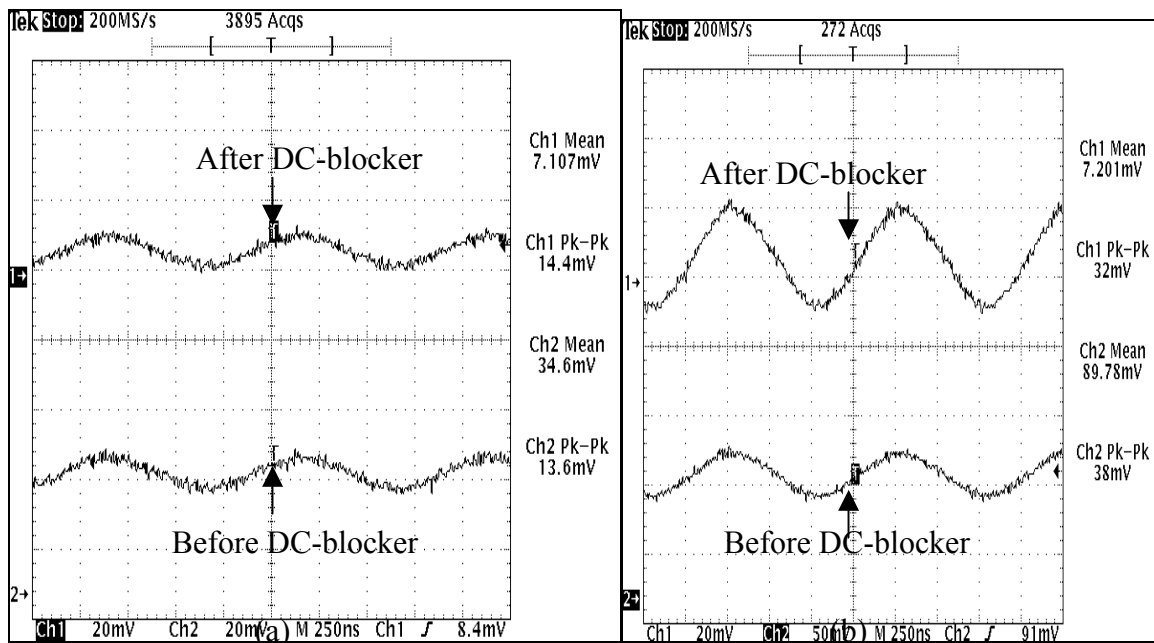
**Figure 36 Experiment Setup for testing DC-blocker**

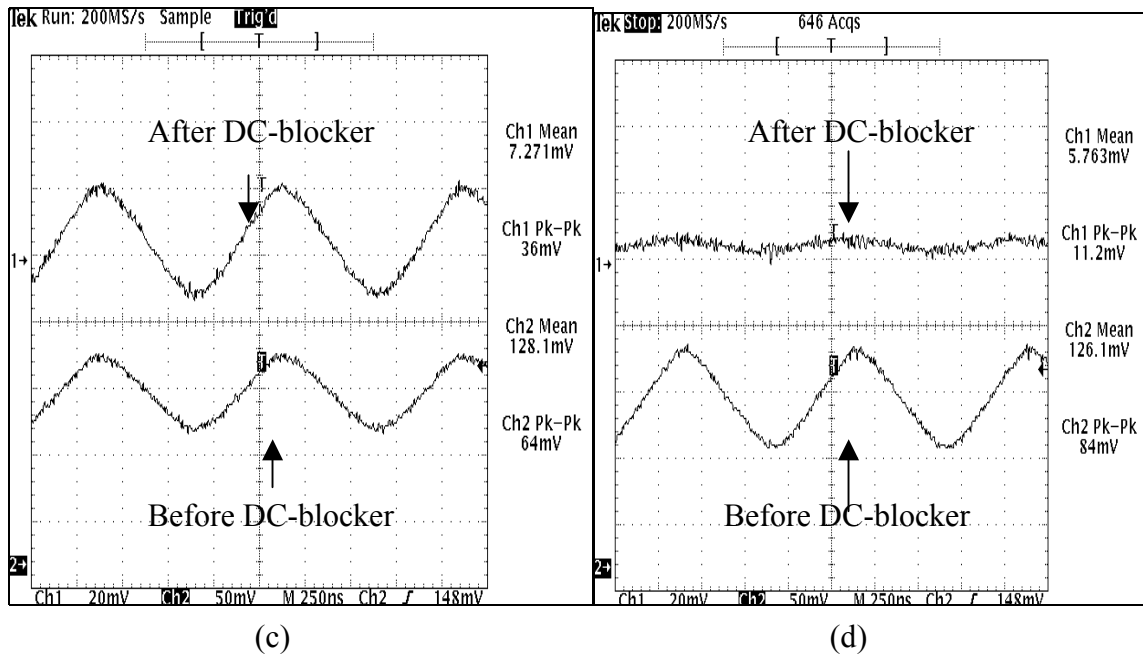
The experiment are summarized in Table 7:

Capacitor Value	Signal After first Stage		Signal After DC blocker and Unity Gain Amplifier		Sinking Time
	Vpp (mV)	Vmean(mV)	Vpp (mV)	Vmean (mV)	
10 pF	84	126.1	11.2	5.763	NA
330 pF	64	128.1	36	7.2	NA
1 nF	25.6	61.8	20	6.5	NA
100 nF	38	89.8	32	7.2	0.8s
4.7 uF	40	101.2	40	7.1	23s

**Table 7 Experiment Result for testing DC-Blocker**

Capacitor of value more than 1nF has successfully brings the mean value down to a fixed value around 7mV (Figure 37 (a)-(b)). When using capacitance value of 330 pF and 10 pF, the signal peak to peak amplitude is suppressed (Figure 37(c)-(d))0





**Figure 37 DC blocker Voltage Waveform when capacitor of value (a) 4.7uF (b) 100nF (c) 330 pF (d) 10 pF**

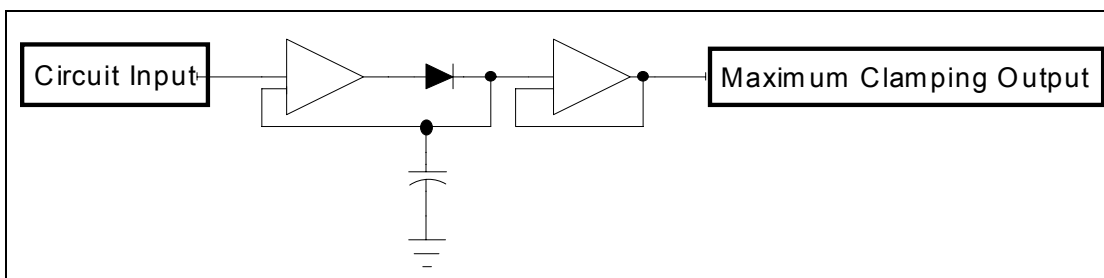
However, it is observed that with high capacitance, DC blocker doesn't establish equilibrium state immediately. For capacitance of 4.7 uF, it took approximately 23 second for the mean value to sink from 23.66mV to 7.2 mV. And for capacitance of 100 nF, it takes around 0.8s for mean value to sink from 14mV to 7mV. This is because the larger the capacitance, the longer time it would take to charge up. This effect would be crucial to our receiver design since one second of incorrect threshold will lead to error of 3 million bits. Also, the transmitter will not be sending signal all the time. Every time a package of data is received, DC blocker will have to take a while to sink to equilibrium value. Although for capacitance less than 100nF, there is no observable sinking time by human eye. However, The actual sinking time will still be significant to one bit period. Consequently, a DC blocker would not be suitable for high speed communication system.

### 5.3.9 Clamping Circuits: To determine a precise threshold value

One other solution for obtaining threshold level is to obtain the maximum value and minimum value of receiving signal. This can be done by building a maximum clamping circuit and a minimum clamping circuit.

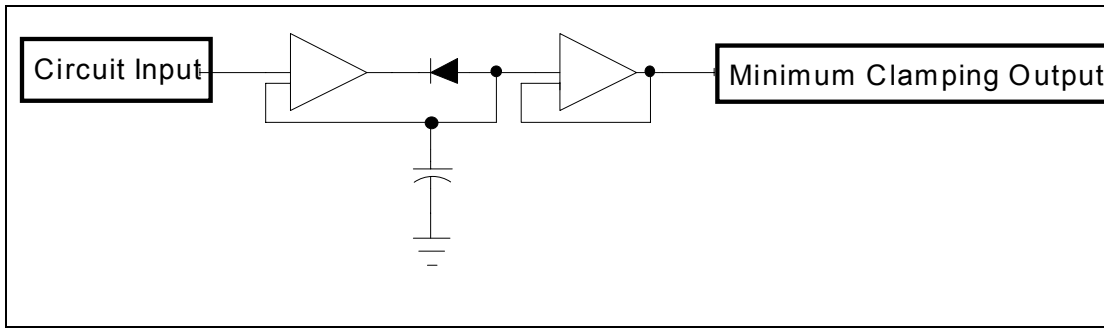
### 5.3.10 Design of Clamping Circuit

The key components of a maximum clamping circuit are the capacitor and diode. The capacitor will act as a “memory” which stores the maximum voltage level, and the diode will act as a “comparing device” which will pass the voltage level into “memory” only when the voltage level is larger than the one stored in capacitor. In order to avoid loading effect, two unity-gain amplifiers are built before and after diode and capacitor (Figure 38).



**Figure 38 Circuit schematics for clamping circuit which obtain the maximum input level**

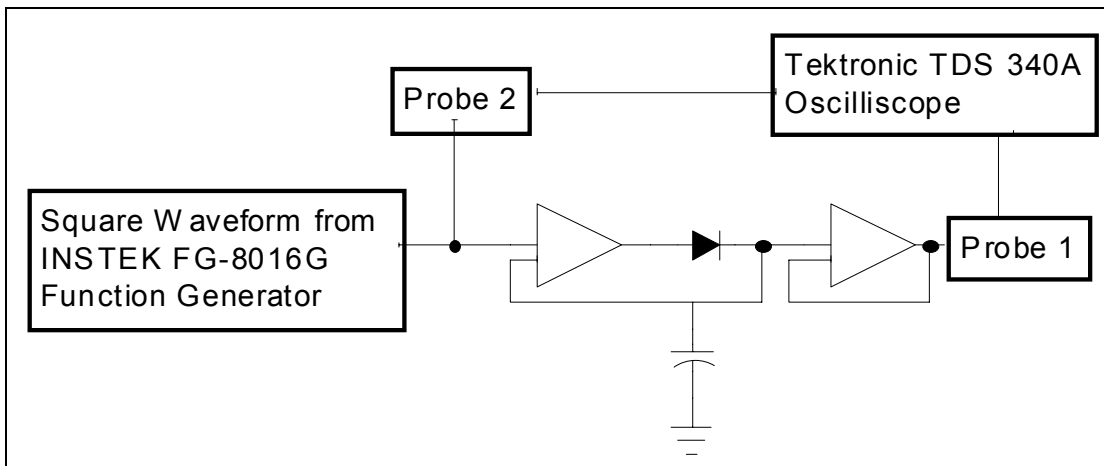
The minimum clamping circuit will be built in the way similar to maximum clamping circuit only the diode is reversed (Figure 39).



**Figure 39** Circuit schematics for clamping circuit which obtain the minimum input level

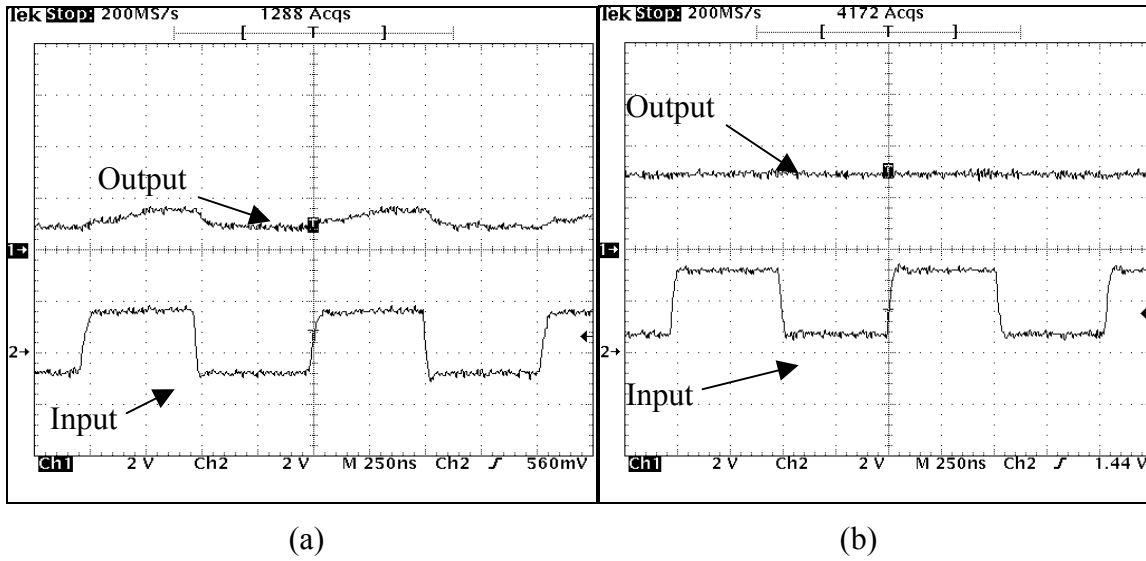
### 5.3.11 Testing Clamping Circuits

The clamping circuits are successfully tested on breadboard in the following test setup.



**Figure 40** Test setup for testing clamping maximum clamping circuit

Capacitors of different values are also tried in the clamping circuit. When the capacitance is low, there is fluctuation in the output of clamping circuit (Figure 41 (a)). The clamping circuit is working properly when the capacitance is larger than 200pF (Figure 41 (b)).



**Figure 41 Clamping circuit input and output when capacitance of (a) 10pF (b) 330pF**

Experiment results of maximum clamping circuit with capacitance of 330 pF are summarized in Table 8.

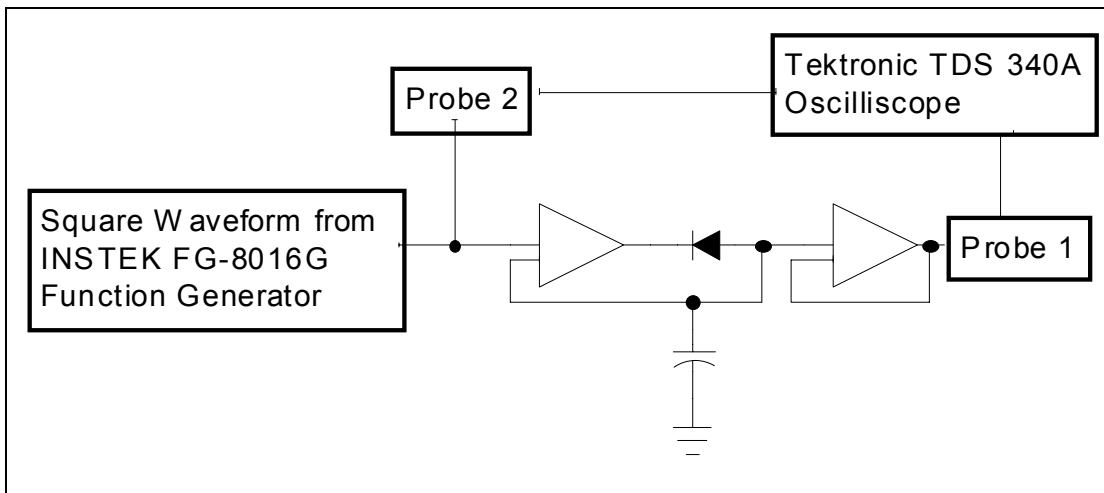
Input Signal Maximum	Output	Error(%)
4.92	3.38	31.30
4.12	3.44	16.51
3.68	3.5	4.89
3.4	3.14	7.64
3.16	2.96	6.32
2.96	2.92	1.35
2.6	2.48	4.61
1.48	1.36	8.11
0.44	0.41	6.82
0.36	0.34	5.56
0.16	0.132	17.5
0.08	0.02	75

**Table 8 Experiment results for maximum clamping circuit**

From the table, we can see that the clamping circuit works quite accurately (within 10% of error), from the range of 200mV to 4.0V. However, the output of clamping circuit

become around 3.8 when the maximum input voltage higher than 4.0 V. This is because that at this amplitude, the voltage is close to the supply voltage of amplifier. The output also become in accurate when the voltage is lower than 200mV since the voltage is too small drive through diode. It is also notice that the clamping circuit doesn't work for negative voltage because the nature of diode.

The minimum clamping circuit is also tested in the same setup (Figure 42) as the maximum circuit. The results are summarized in Table 9.



**Figure 42 Test setup for testing clamping maximum clamping circuit**

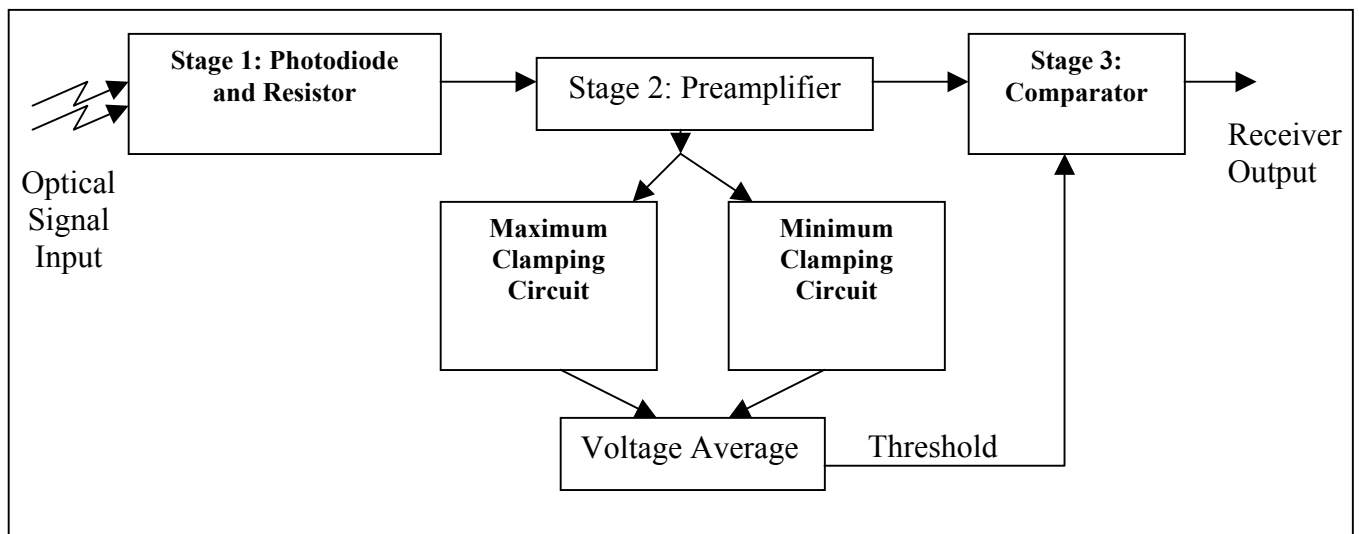
Input Signal Minimum	Output of Clamping Circuit
3.52	5.12
3.44	3.48
2.8	2.82
1	1.1
0.32	0.36

**Table 9 Experiment results for minimum clamping circuit**

The minimum clamping circuit works well. However, it is noticed that when the maximum of input signal exceeds 5V (which is also the value of  $V_{cc}$ ). The output will become constantly 5.

### 5.3.12 Overall Receiver System

The overall receiver system is built in the topology shown in Figure 43.



**Figure 43 Overall receiver system**

The overall system is tested with stage 1 to stage 3 build on vector board and clamping circuit on breadboard. The receiver produce satisfactory output waveform as in Figure 34.

The dynamic range is larger than receiver with fixed threshold. It is working properly from 0 cm to 4 cm. The range is basically limited by operational range of clamping circuits.

## **6 System Implementation**

Throughout the year, we have tried several methods to test our design at different stages. We started implementing infrared transceiver design with the breadboard, then we moved to using the micro strip when dealing with USB transceiver design, and finally we implemented the USB-IR control system with the PCB (printed circuit board). Different implementations have different advantages and disadvantages. This section will discuss about every implementation, but however it will mainly focus on PCB and its manufacturing procedures.

### ***6.1 Breadboard***

Breadboard is the most commonly used platform for circuit testing. It is very simple and convenient however, sometimes it doesn't provide good results when working with high-speed signals due to bad connections, interference and noise. We tested our IR transceiver and we found a lot of signal distortion due to high frequency signals being used. Some of the problems were temporary solved by using two separate boards but some of the connection problem still existed.

### ***6.2 Micro-strip Circuit Board***

Micro-strip circuit board was being used during the testing of the USB transceivers. The USB transceiver chip obtained was SOP (small-outline package) and surface mounted. Those chips couldn't be used on the breadboard and there isn't any socket available for

them on the market. Since high speed signals would be used and we would like to reduce the chances of getting wired tangled together to reduce interferences, we came up with the idea of placing the chip on a board and using micro-strip to connect components together. However, using micro strip created more problems then expected.

First of all, the width of the leg of a chip is about 0.2 mm whereas the width of a micro-strip is 2 mm. Second, to ensure relatively good connection, the leg of the chip needs to be soldered onto the micro strip. Therefore, basically we needed to solder a piece of a wire in order to connect the chip and micro strip together. That created a lot of problems afterward since the heat from the solder gun melt the glues on the back side of the micro-strip then the micro strip didn't stay flat on the board anymore. Also, the leg of the chip became very brittle after being heated up and coated with solder. It wasn't an easy task to solder the wire and the chip together. The final testing demo didn't look like the expected. Nothing was sitting on the board flat but rather floating and dancing in the air. It was rather challenging to carry the whole board around since it needed a lot of care to prevent it from breaking up to pieces.

### **6.3 PCB (*Printed Circuit Board*)**

Our final design was implemented on a PCB. We decided to do so since most components we acquired were surface mounted and small. There were also other considerations such as PCB eliminates many problems we were facing. Using PCB, the connections on the board are secured and not loosely like what breadboard does and the

board size is small and compact. It will make the testing easier when all those problems are no longer there.

The process of making PCB can generally be divided into designing and implementation stages. The process of designing involves drawing the layout traces that connect the individual components together. Then, the process of implementation involves printing the copper traces onto the board.

### **6.3.1 Designing**

The process of designing becomes very complicated as the complexity of connections on the board increases. The complexity increases dramatically especially when it is a single layer PCB. Most of the time, PCB design works are done through CAD (Computer Aided Design) software. We had tried AutoCAD which a very popular but general all-purpose software. We could use AutoCAD to design the layout but it was a time consuming task to draw the footprint of every component we used. We also tried Winboard PCB, a PCB specified CAD software. Winboard PCB is the product of Ivex Design International (<http://www.ivex.com>) and there is free demo software on the website. The PCB specified CAD software provides a library of components layout that saved us a lot of time and work.

We used a single layer PCB and even though we didn't have many components, we still ran into connection crossover problem. We minimized the number of crossover connections by re-assigning pin functions on the Altera EPM7128S programmable logic

device and somewhat bypassed the problem by connecting the crossover connection with a thin wire.

### **6.3.2 Implementation**

There are a number of ways to make PCB prototype after the layout design is done. It can be sent to a PCB manufacturing company and have the prototype professionally done or there are DIY (Do-It-Yourself) tools that help you to manufacture PCB prototypes on your own. A few popular methods are on the market and these include the peelable masking method (Press-n-Peel method) and UV solder masking method. UV solder masking method needs a UV light source for good results. Even though an ordinary fluorescent light can be used but the process of developing will take about 10 times longer. The copper board used for UV solder masking is generally more expensive than the copper board used for the press-n-peel method. We decided to go with the Press-n-Peel method that was the simplest and required least accessories.

The Press-n-Peel method takes a transfer film to transfer the layout design onto the copper board. The copper board is then etched with ferric chloride ( $\text{FeCl}_3$ ) to take out excess copper and then scrubbed with steel wool to reveal the copper traces.

The directions and details of Press-n-Peel method are going to be discussed afterward. Below, it is a short summary of how to make a good PCB with Press-n-Peel and details need to be care of.

- Place a piece of paper between the board and the clothes iron to reduce the chance of scratching the iron surface.
- Press firmly when ironing the transfer film onto the board. Rotate and adjust position of the clothes iron frequently to ensure that the copper board and the transfer film have a flat and firm contact with the clothes iron.
- Trim the copper board to minimize the etching time and unnecessary etching of copper.
- When ironing the Press-n-Peel onto the board, if the image is successfully transfer onto the board, there will be traces reflected off the plastic shiny side of the transfer film. If it is not successful, the image can be ironed on again. If it still can't be successfully transfer onto the board, a permanent marker can be used to fulfill the missing traces.
- We used ferric chloride ( $\text{FeCl}_3$ ) to etch copper. Even though it does not produce dangerous fumes, is odourless and, though corrosive, is not absorbed through the skin, it is still a good idea to wear plastic glove.
- Do not etch the copper board more than 1 ½ hours because the longer etching time, the chance that the traces on the board get dissolved. The film then detaches from the board and the copper got etched. It will produce a bad resolution.
- Keep the etching solution warm will increase the etching rate.

### 6.3.3 Procedures / Directions

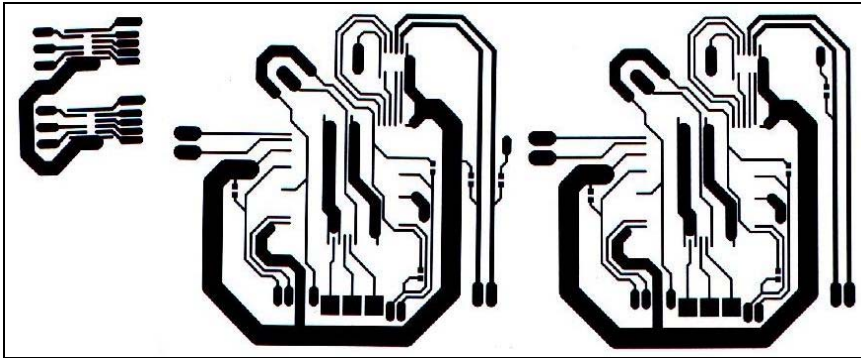
Prepare: Clothes Iron, Steel Wool #00 (Superfine), liquid soap, Photocopy or Laser

Printed Circuit Image, & Directions, ferric chloride, utility knife

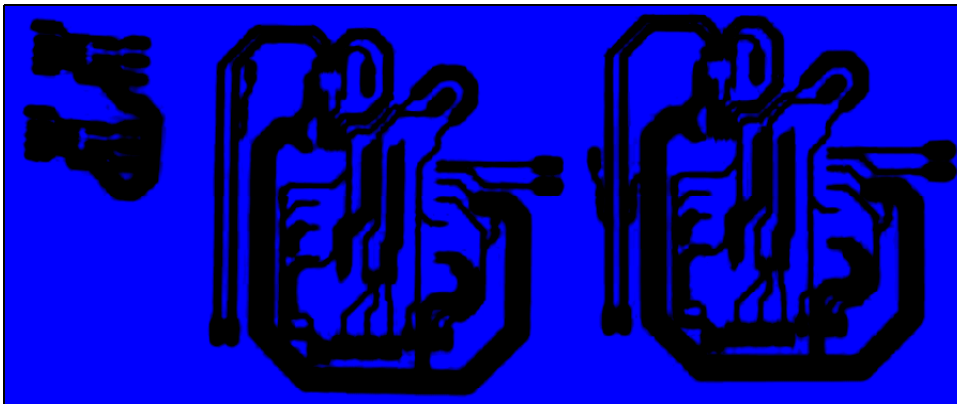
Figures are listed at the end of 6.3.3

1. Refer to Figure 44 and Figure 45. Photocopy or Laser Print circuit image onto the dull side (emulsion) of Press-n-Peel Image Transfer Film. When using photocopy, remember to photocopy as photocopying transparencies.
2. Cut Press-n-Peel, leaving a 0.5 cm border around the circuit image. Trim the board to size.
3. Clean copper board with steel wool. Remove oxides from the surface and rinse cleaned board with soap and water. Be sure to remove all soap residues. Dry thoroughly. Be sure to scrape any burrs that appear on the edge of the board that may have resulted from the cutting/shearing process. Burrs tend to keep the iron from making solid contact with the Press-n-Peel Film.
4. Refer to Figure 46. Place Press-n-Peel with image face down onto clean copper board. Iron the Press-n-Peel Film to the board. Place a piece of plain paper between the iron and the film to reduce friction. **DO NOT USE STEAM!**  
  
Temperature setting on the iron is critical, and dependent upon your laser printer or photocopier. Suggested starting temperature is 275-325 degrees F. Iron settings generally are between the "acrylic" and "polyester" settings. Time varies with the size and thickness of the board. Generally this is 3-4 minutes.
5. Refer to Figure 47. Let the board cool off and peel the film off.

6. Refer to Figure 48. Trim the board (if necessary) to the final size. Wash the board in soap & water before etching to remove surface oxidation. Etch with any standard copper board etching solution -- Ferric Chloride, Ammonium Persulfate, etc.
7. Refer to Figure 49 and Figure 50. Using steel wool, scrub the Press-n-Peel image off as to reveal copper traces. This is best done under running water.



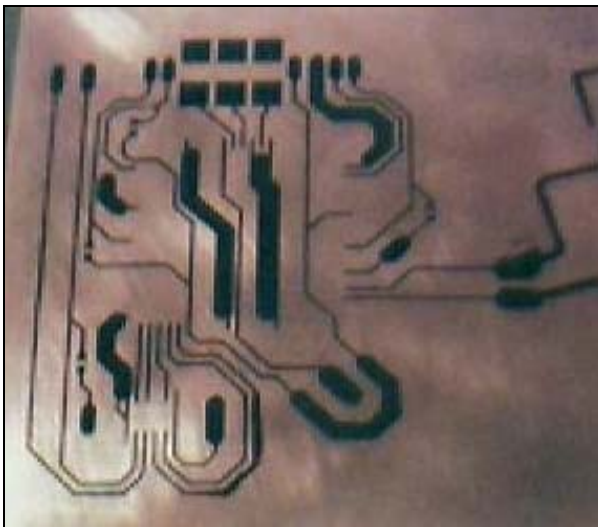
**Figure 44 Design layout of the USB-IR bridge controller**



**Figure 45 Design layout photocopied on the Press-n-Peel film**



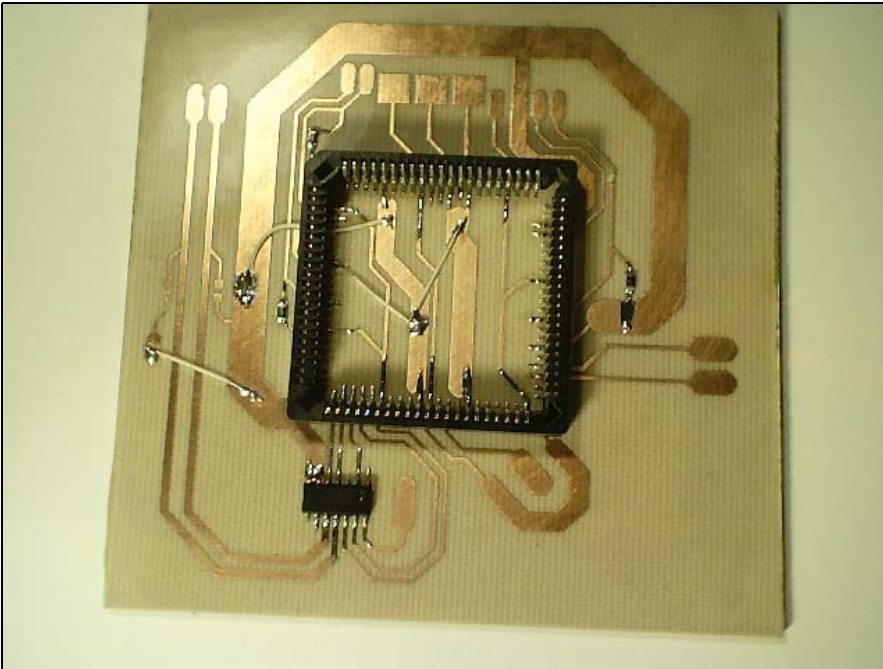
**Figure 46 Use clothes iron to transfer the image onto the board**



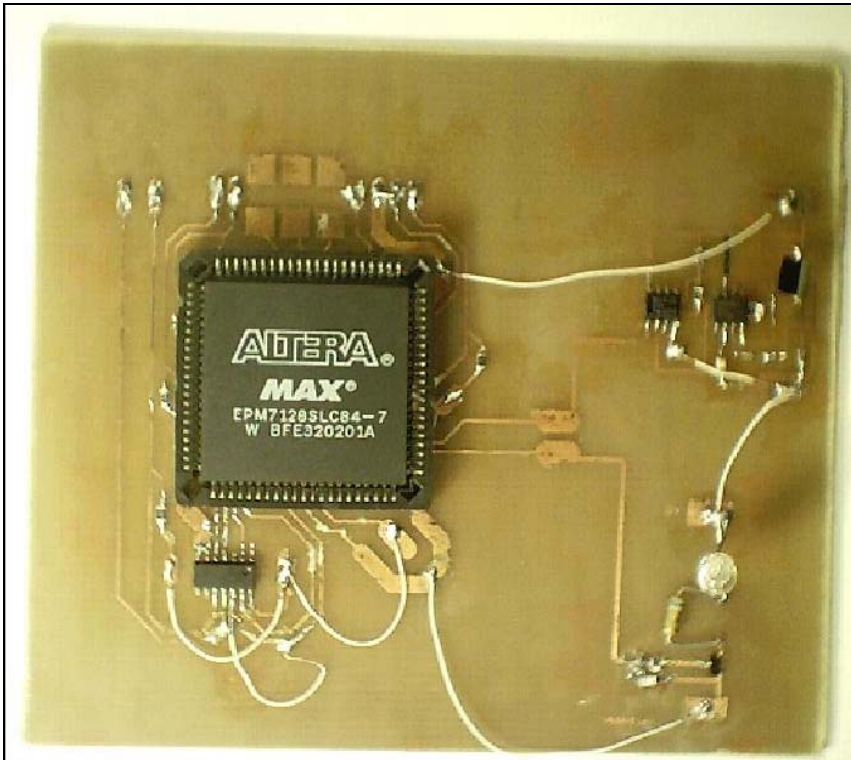
**Figure 47 Image of the trace on the copper board**



**Figure 48 Etch the copper board with Ferric Chloride**



**Figure 49 PCB after etching. Socket and wires been soldered on**



**Figure 50 USB-IR test board – first attempt**

## 7 Conclusion

We had successfully completed our interface design. However, during the final testing with an USB mouse, the interface didn't work as expected. We had not yet figured out what went wrong because we didn't have enough time to do so. Before we proceeded with the final testing, we had successfully finished the USB-IR controller simulation and the results showed that the algorithm works fine.

### 7.1 Cost Estimate

USB devices are so popular and its cost of implementation is very low. Our design minimizes a lot of unnecessary cost by keeping the design simple and efficient. A lot of functions normally required extra chip sets have been incorporated into the Altera EPM7128S programmable logic device. Also we had ran through some calculations summarized in Table 10 and market research for how enthusiastic consumers are willing to pay. We reduce our product to be less than \$50.00 and still profitable.

Estimated Cost of Overall System	
<b>USB Transceiver</b>	\$0.20
Infrared Transmitters	
Fairchild QSD224 LED's (x2)	\$0.10
Fairchild 2N7002 MOSFET's (x2)	\$0.10
Resistors (x6)	\$0.06
<b>Infrared Receivers</b>	
Fairchild QSE773 Photodiodes (x2) MAXIM MAX473	\$0.92
Op Amps (x2)	\$2.90
MAXIM MAX907 Comparators (x2)	\$3.40
Resistors (x10)	\$0.10
Capacitors (x5)	\$0.25

<b>USB-IR Bridge System</b>	
Programmable Logic Device	\$15.00
<b>Printed Circuit Boards Development Cost</b>	\$5.00
<b>Total Project Cost</b>	\$35.0

**Table 10 Estimated Cost of Overall System**

## ***7.2 Future development***

Even though this final testing wasn't successfully completed, we believe that our design is just one step away from functioning. We have already focus on its future development and ready to continue to improve this system as soon as we complete the final test and resolve the problems facing currently.

We eye on an IR transceiver's bit rate improvement to match the current high-speed USB need. Also, incorporate with a hub to create a multi-device system and construct the system with better PCB manufacturing technology to make smaller PCB. Moreover incorporate the host adaptor into laptop's infrared port. This will benefit the laptop users to connect to all the USB devices through infrared.

## Reference

---

- [1] J. Pappas, “USB: Past, Present and Future,” presented at the USB Developer’s Conference, 1999
- [2] M. Slater, “The Role of Universal Serial Bus,” presented at the USB Developer’s Conference, 1995
- [3] Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, pp.18-20
- [4] Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, p.17
- [5] E. Barsotti, S. Zimmermann, “Low Current Differential Signals,” Document #ESE-SVX-950605, Fermi National Accelerator Laboratory, 1996
- [6] J. Hyde, USB Design by Example: A Practical Guide to Building I/O Devices, Wiley Computer Publishing, 1999, p.44
- [7] Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, p.155
- [8] Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, pp.155-156
- [9] J. Hyde, USB Design by Example: A Practical Guide to Building I/O Devices, Wiley Computer Publishing, 1999, p.19
- [10] Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, pp.133-134
- [11] USB1T11A Universal Serial Bus Transceiver Data Sheet, Fairchild Semiconductor, 1999

- 
- <sup>[12]</sup> Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, pp.140-141
- <sup>[13]</sup> J. Hyde, USB Design by Example: A Practical Guide to Building I/O Devices, Wiley Computer Publishing, 1999, p.23
- <sup>[14]</sup> Universal Serial Bus Specification, Revision 1.1, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, NEC Corporation, 1998, pp.171-172
- <sup>[15]</sup> J. Hyde, USB Design by Example: A Practical Guide to Building I/O Devices, Wiley Computer Publishing, 1999, p.39
- <sup>[16]</sup> Encyclopedia of Electronic Circuits, Version 1 to Version 7, Rudolf F. Graf, 1985-1999
- <sup>[17]</sup> Infrared Data Association Serial Infrared Physical Layer Specification, Version 1.3, IrDA, 1998
- <sup>[18]</sup> Optoelectronics Circuits Manual, Version 2, R M Marston, 1999
- <sup>[19]</sup> Govind P. Agrawal, Fiber-Optic Communication Systems, Second Edition, Wiley-Interscience Publishing, 1997, pp. 163-185

# Appendices

## A Host Modulator

%%VHDL code for the host modulator

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY hstenc IS
    PORT( clk: IN STD_LOGIC;
          reset: IN STD_LOGIC;
          d2: IN STD_LOGIC;          %D- input
          d1: IN STD_LOGIC;          %D+ input
          oe: OUT STD_LOGIC;         %output enable request
          s: OUT STD_LOGIC);         %serial bit output
END hstenc;
```

```
ARCHITECTURE Behavior OF hstenc IS
    Type state_name IS (s8, s7, s6, s5, s4, s3, s2,
                        i0, i1, i2, i3, k0, k1, j0, j1);
    Signal state: state_name;
BEGIN Process(reset, clk)
    BEGIN IF reset='0' THEN state<=s8;
          ELSIF(clk'EVENT AND clk='1') THEN
              CASE state IS
```

%%States s2 through s8 counts consecutive SE0's.  
%%State s8 is the initialization state (disconnected device).  
%%Device is deemed disconnected for count longer than 4 bit times at s8.  
%%For a disconnected device, OE request is disabled on the host side and  
%% enabled on the device side.  
%%(This step is a redundant assurance, as the device detachment is  
%% detected by the demodulator, and the direction controller ensures the  
%% correct data direction at this situation.)

```
        WHEN s8=>
            s<='0';
            oe<='0';
            IF d2='0' AND d1='0' THEN state<=s8;
            ELSE state<=i0;
            END IF;
        WHEN s2=>
            s<='0';
            oe<='0';
            IF d2='0' AND d1='0' THEN state<=s3;
            ELSE state<=i0;
            END IF;
        WHEN s3=>
            s<='0';
            IF d2='0' AND d1='0' THEN state<=s4;
            ELSE state<=i0;
            END IF;
        WHEN s4=>
            s<='0';
            IF d2='0' AND d1='0' THEN state<=s5;
            ELSE state<=i0;
            END IF;
        WHEN s5=>
```

```

s<='0';
    IF d2='0' AND d1='0' THEN state<=s6;
    ELSE state<=i0;
    END IF;
WHEN s6=>
s<='0';
    IF d2='0' AND d1='0' THEN state<=s7;
    ELSE state<=i0;
    END IF;
WHEN s7=>
s<='0';
    IF d2='0' AND d1='0' THEN state<=s8;
    ELSE state<=i0;
    END IF;

```

%%States i0 through i3 counts consecutive idle J-states.  
%%OE request is disabled at second idle state counted at i3.

```

WHEN i0=>
s<='1';
    IF d2='0' AND d1='0' THEN state<=s2;
    ELSE state<=i1;
    END IF;
WHEN i1=>
s<='0';
    IF d2='1' AND d1='0' THEN state<=i2;
    ELSIF d2='0' AND d1='1' THEN state<=k0;
    ELSE state<=s2;
    END IF;
WHEN i2=>
s<='1';
    IF d2='0' AND d1='0' THEN state<=s2;
    ELSE state<=i3;
    END IF;
WHEN i3=>
oe<='0';
s<='0';
    IF d2='1' AND d1='0' THEN state<=i2;
    ELSIF d2='0' AND d1='1' THEN state<=k0;
    ELSE state<=s2;
    END IF;

```

%%State k0 and k1 are reached when a K-state is detected.  
%%OE request is enabled.  
%%Outputs serial bit stream "01."

```

WHEN k0=>
s<='0';
oe<='1';
    IF d2='0' AND d1='0' THEN state<=s2;
    ELSE state<=k1;
    END IF;
WHEN k1=>
s<='1';
    IF d2='1' AND d1='0' THEN state<=j0;
    ELSIF d2='0' AND d1='1' THEN state<=k0;
    ELSE state<=s2;
    END IF;

```

%%State j0 and j1 are reached when a J-state is detected.  
%%OE request is enabled.  
%%Outputs serial bit stream "10."

```

        WHEN j0=>
            s<='1';
            oe<='1';
            IF d2='0' AND d1='0' THEN state<=s2;
            ELSE state<=j1;
            END IF;
        WHEN j1=>
            s<='0';
            IF d2='1' AND d1='0' THEN state<=j0;
            ELSIF d2='0' AND d1='1' THEN state<=k0;
            ELSE state<=s2;
            END IF;

        END CASE;
    END IF;
END PROCESS;
END Behavior;

```

## **B Host Demodulator**

%%VHDL code for the host demodulator

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY hstdec IS
    PORT( clk: IN STD_LOGIC;
          reset: IN STD_LOGIC;
          s: IN STD_LOGIC;           %serial bit input
          oe: OUT STD_LOGIC;        %output enable request
          dsc: OUT STD_LOGIC;       %device disconnection notice
          d2: OUT STD_LOGIC;        %D- output
          d1: OUT STD_LOGIC);       %D+ input
END hstdec;

ARCHITECTURE Behavior OF hstdec IS
    Type state_name IS (s8, s7, s6, s5, s4, s3, s2,
                        i0, i1, i2, i3, k0, k1, j0, j1);
    Signal state: state_name;
BEGIN Process(reset, clk)
    BEGIN IF reset='0' THEN state<=s8;
          ELSIF(clk'EVENT AND clk='1') THEN

%%States s2 through s8 counts consecutive SE0's.
%%OE request is disabled immediately after a SE0 is detected.
%%State s8 is the initialization state (disconnected device).
%%Device is deemed disconnected for count longer than 4 bit times at s8.
%%For a disconnected device, disconnection notice (active low) is sent.

            CASE state IS
                WHEN s8=>
                    d2<='0';
                    d1<='0';
                    dsc<='0';
                    IF s='0' THEN state<=s8;
                    ELSE state<=i0;
                    END IF;
                WHEN s2=>
                    d2<='0';
                    d1<='0';
                    oe<='0';
                    IF s='0' THEN state<=s3;
                    ELSE state<=i0;
                    END IF;
                WHEN s3=>
                    IF s='0' THEN state<=s4;
                    ELSE state<=i0;
                    END IF;
                WHEN s4=>
                    d2<='0';
                    d1<='0';
                    IF s='0' THEN state<=s5;
                    ELSE state<=i0;
                    END IF;
                WHEN s5=>
                    IF s='0' THEN state<=s6;
                    ELSE state<=i0;
                    END IF;
            END CASE;
        END IF;
    END IF;
END Process;
```

```

WHEN s6=>
d2<='0';
d1<='0';
    IF s='0' THEN state<=s7;
    ELSE state<=i0;
    END IF;
WHEN s7=>
    IF s='0' THEN state<=s8;
    ELSE state<=i0;
    END IF;

```

%%States i0 through i3 counts consecutive idle J-states.  
%%OE request is disabled at second idle state counted at i3.  
%%Disconnection notice is turned off at i0.

```

WHEN i0=>
dsc<='1';
    IF s='0' THEN state<=i1;
    ELSE state<=s2;
    END IF;
WHEN i1=>
d2<='1';
d1<='0';
    IF s='1' THEN state<=i2;
    ELSE state<=k0;
    END IF;
WHEN i2=>
    IF s='0' THEN state<=i3;
    ELSE state<=s2;
    END IF;
WHEN i3=>
d2<='1';
d1<='0';
oe<='0';
    IF s='1' THEN state<=i2;
    ELSE state<=k0;
    END IF;

```

%%State k0 and k1 are reached when a bit sequence "01" is detected.  
%%OE request is enabled.  
%%Outputs differential K-state.

```

WHEN k0=>
oe<='1';
    IF s='1' THEN state<=k1;
    ELSE state<=s2;
    END IF;
WHEN k1=>
d2<='0';
d1<='1';
    IF s='1' THEN state<=j0;
    ELSE state<=k0;
    END IF;

```

%%State j0 and j1 are reached when a bit sequence "10" is detected.  
%%OE request is enabled.  
%%Outputs Differential J-state.

```

WHEN j0=>
oe<='1';
    IF s='0' THEN state<=j1;
    ELSE state<=s2;
    END IF;

```

```
        WHEN j1=>
          d2<='1';
          d1<='0';
          IF s='1' THEN state<=j0;
          ELSE state<=k0;
          END IF;

        END CASE;
      END IF;
    END PROCESS;
  END Behavior;
```

## **C Host Data Direction Controller**

*%% VHDL code for the host data direction controller*

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY hstdr IS
    PORT( clk: IN STD_LOGIC;
          reset: IN STD_LOGIC;
          oem: IN STD_LOGIC;           %output enable request of modulator
          oed: IN STD_LOGIC;          %output enable request of demodulator
          ds: IN STD_LOGIC;           %disconnection notice
          ddc: OUT STD_LOGIC);        %control signal
END hstdr;

%%ddc=1 enables modulator, disables demodulator.
%%    allows input data from differential bus lines.
%%    blocks input data from infrared receiver port.
%%ddc=0 enables demodulator, disables modulator.
%%    allows input data from infrared receiver port.
%%    blocks input data from differential bus lines.

ARCHITECTURE Behavior OF hstdr IS
    Type state_name IS (dsc, def, m, d);
    Signal state: state_name;
BEGIN Process(reset, clk)
    BEGIN IF reset='0' THEN state<=dsc;
          ELSIF(clk'EVENT AND clk='1') THEN
            CASE state IS

%%State dsc is the initialization state (disconnected device).
%%Allows input data from the device side, such that the host can
%% detect the disconnection.
%% ddc=0 for the host controller
%% ddc=1 for the device controller
%%Moves on to default state, when the disconnection notice is inactive.
                WHEN dsc=>
                    ddc<='0';
                    IF ds='1' THEN state<=def;
                    ELSE state<=dsc;
                    END IF;

%%At default state, modulator is enabled.
%%Moves on to state m, when OE request from modulator is enabled.
%%Moves on to state d, when OE request from demodulator is enabled
%%    AND OE request from modulator is disabled.
                WHEN def=>
                    ddc<='1';
                    IF oem='0' AND oed='0' AND ds='1' THEN state<=def;
                    ELSIF oem='1' AND oed='1' AND ds='1' THEN state<=m;
                    ELSIF oem='1' AND oed='0' AND ds='1' THEN state<=m;
                    ELSIF oem='0' AND oed='1' AND ds='1' THEN state<=d;
                    ELSE state<=dsc;
                    END IF;

%%At m state, modulator is enabled, and demodulator is disabled.
                WHEN m=>
                    ddc<='1';
```

```

        IF oem='1' AND ds='1' THEN state<=m;
        ELSIF oem='0' AND ds='1' THEN state<=def;
        ELSE state<=dsc;
        END IF;

%%At d state, demodulator is enabled, and modulator is disabled.
    WHEN d=>
        ddc<='0';
        IF oed='1' AND ds='1' THEN state<=d;
        ELSIF oed='0' AND ds='1' THEN state<=def;
        ELSE state<=dsc;
        END IF;
    END CASE;

    END IF;
END PROCESS;
END Behavior;

```

## ***D Sampler for Infrared Input***

%%VHDL code for the sampler

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY samp16 IS
    PORT( clk: IN STD_LOGIC;
          i: IN STD_LOGIC;      %sampler input
          y: OUT STD_LOGIC);    %sampler output
END samp16;
```

%%Samples the infrared serial input at 16X USB data rate (24Mbps)

```
ARCHITECTURE Behavior OF samp16 IS
    Type state_name IS (ini, h1, h2, h3, h4, h5, h6, h7, h8,
                        11, 12, 13, 14, 15, 16, 17, 18);
    Signal state: state_name;
BEGIN Process(clk)
    BEGIN IF(clk'Event AND clk='1') THEN
        CASE state IS
```

%%Initialization state

```
        WHEN ini=>
            y<='0';
            IF i='0' THEN state<=11;
            ELSE state<=h1;
            END IF;
```

%%Counts the number of consecutive data low samples.

%%Outputs signal low at the fourth sampling period.

%%Glitches, signals lasting shorter than 4 periods, are ignored.

```
        WHEN 11=>
            IF i='0' THEN state<=12;
            ELSE state<=h1;
            END IF;
        WHEN 12=>
            IF i='0' THEN state<=13;
            ELSE state<=h1;
            END IF;
        WHEN 13=>
            IF i='0' THEN state<=14;
            ELSE state<=h1;
            END IF;
        WHEN 14=>
            y<='0';
            IF i='0' THEN state<=15;
            ELSE state<=h1;
            END IF;
        WHEN 15=>
            IF i='0' THEN state<=16;
            ELSE state<=h1;
            END IF;
        WHEN 16=>
            IF i='0' THEN state<=17;
            ELSE state<=h1;
            END IF;
        WHEN 17=>
```

```

        IF i='0' THEN state<=l8;
        ELSE state<=h1;
        END IF;
    WHEN l8=>
        IF i='0' THEN state<=l1;
        ELSE state<=h1;
        END IF;

%%Counts the number of consecutive data high samples.
%%Outputs signal high at the fourth sampling period.
%%Glitches, signals lasting shorter than 4 periods, are ignored.
    WHEN h1=>
        IF i='1' THEN state<=h2;
        ELSE state<=l1;
        END IF;
    WHEN h2=>
        IF i='1' THEN state<=h3;
        ELSE state<=l1;
        END IF;
    WHEN h3=>
        IF i='1' THEN state<=h4;
        ELSE state<=l1;
        END IF;
    WHEN h4=>
        y<='1';
        IF i='1' THEN state<=h5;
        ELSE state<=l1;
        END IF;
    WHEN h5=>
        IF i='1' THEN state<=h6;
        ELSE state<=l1;
        END IF;
    WHEN h6=>
        IF i='1' THEN state<=h7;
        ELSE state<=l1;
        END IF;
    WHEN h7=>
        IF i='1' THEN state<=h8;
        ELSE state<=l1;
        END IF;
    WHEN h8=>
        IF i='1' THEN state<=h1;
        ELSE state<=l1;
        END IF;

    END CASE;
    END IF;
END PROCESS;
END Behavior;

```

## ***E Original Task Assignment***

<b>Tasks</b>	<b>Date of Completion</b>	<b>Responsibility</b>
Proposal	October 19, 2001	Hou, Hsiao, Hung
Hardware Development <ul style="list-style-type: none"> <li>▪ Specifications Research               <ul style="list-style-type: none"> <li>○ USB</li> <li>○ IR</li> </ul> </li> <li>▪ Schematic Design               <ul style="list-style-type: none"> <li>○ USB-IR Bridge</li> <li>○ IR Transceiver</li> <li>○ Overall Circuit</li> </ul> </li> <li>▪ Component Procurement</li> <li>▪ Hardware Implementation</li> <li>▪ Testing and Debugging</li> <li>▪ Documentation</li> </ul>	October 15, 2001 October 15, 2001  November 15, 2001 December 15, 2001 December 15, 2001 October 31, 2001 January 31, 2002 February 15, 2002 February 31, 2002	Hsiao Hou  Hung Hou Hsiao Hung Hsiao Hung Hou, Hsiao, Hung
Software Development <ul style="list-style-type: none"> <li>▪ Specifications Research               <ul style="list-style-type: none"> <li>○ USB</li> <li>○ IR</li> </ul> </li> <li>▪ USB Driver Development</li> <li>▪ Firmware Development               <ul style="list-style-type: none"> <li>○ USB</li> <li>○ IR</li> <li>○ USB-IR Bridge</li> </ul> </li> <li>▪ System Integration</li> <li>▪ Documentation</li> </ul>	October 15, 2001 October 15, 2001 December 15, 2001  January 31, 2002 January 31, 2002 January 31, 2002 February 15, 2002 February 31, 2002	Hsiao Hou Hsiao  Hsiao Hou Hung Hung Hou, Hsiao, Hung
Final Implementation <ul style="list-style-type: none"> <li>▪ Hardware and Software Integration</li> <li>▪ Testing and Debugging               <ul style="list-style-type: none"> <li>○ Testing with a USB mouse</li> <li>○ Testing with a USB scanner</li> <li>○ Testing with a USB HDD</li> </ul> </li> </ul>	March 15, 2002  March 31, 2002 March 31, 2002 March 31, 2002	Hung  Hung Hsiao Hou
Documentation <ul style="list-style-type: none"> <li>▪ Interim Report</li> <li>▪ Final Report</li> </ul>	January 11, 2002 April 12, 2002	Hou, Hsiao, Hung Hou, Hsiao, Hung

## ***F Revised Task Assignment***

<b>Tasks</b>	<b>Date of Completion</b>	<b>Responsibility</b>
Proposal	October 19, 2001	Hou, Hsiao, Hung
<b>Hardware Development</b> <ul style="list-style-type: none"> <li>▪ Specifications Research <ul style="list-style-type: none"> <li>○ USB</li> <li>○ IR</li> <li>○ Power System</li> <li>○ USB-IR Bridge</li> </ul> </li> <li>▪ Schematic Design <ul style="list-style-type: none"> <li>○ IR Transceiver</li> <li>○ USB-IR Bridge</li> <li>○ Overall System</li> </ul> </li> <li>▪ Component Procurement <ul style="list-style-type: none"> <li>○ USB Transceivers</li> <li>○ IR Transceivers</li> <li>○ Power System</li> <li>○ USB-IR Bridge</li> </ul> </li> <li>▪ Module Implementation <ul style="list-style-type: none"> <li>○ USB Transceivers</li> <li>○ IR Transceivers</li> <li>○ Power System</li> <li>○ USB-IR Bridge</li> <li>○ Overall System</li> </ul> </li> <li>▪ Testing and Debugging <ul style="list-style-type: none"> <li>○ USB Transceivers</li> <li>○ IR Transceivers</li> <li>○ Power System</li> <li>○ USB-IR Bridge</li> <li>○ Overall system</li> </ul> </li> <li>▪ Overall hardware System</li> </ul>	<ul style="list-style-type: none"> <li>October 15, 2001</li> <li>October 15, 2001</li> <li>October 15, 2001</li> <li>January 31, 2002</li> <li>December 15, 2001</li> <li>January 31, 2002</li> <li>February 15, 2002</li> <li>January 1, 2002</li> <li>January 1, 2002</li> <li>January 15, 2002</li> <li>January 15, 2002</li> <li>January 15, 2002</li> <li>January 15, 2002</li> <li>January 15, 2002</li> <li>January 15, 2002</li> <li>February 1, 2002</li> <li>February 15, 2002</li> <li>February 28, 2002</li> <li>February 1, 2002</li> <li>February 1, 2002</li> <li>February 1, 2002</li> <li>February 21, 2002</li> <li>March 7, 2002</li> <li>March 22, 2002</li> </ul>	<ul style="list-style-type: none"> <li>Hsiao</li> <li>Hou</li> <li>Hung</li> <li>Hsiao, Hou</li> <li>Hou</li> <li>Hsiao, Hou</li> <li>Hung, Hsiao, Hou</li> <li>Hung</li> <li>Hsiao</li> <li>Hou</li> <li>Hung</li> <li>Hou, Hsiao</li> <li>Hou, Hung, Hsiao</li> <li>Hsiao</li> <li>Hou</li> <li>Hung</li> <li>Hou, Hsiao</li> <li>Hou, Hung Hsiao</li> <li>Hsiao, Hou, Hung</li> </ul>
<b>Software Development</b> <ul style="list-style-type: none"> <li>▪ Specifications Research <ul style="list-style-type: none"> <li>○ USB</li> <li>○ IR</li> </ul> </li> <li>▪ Firmware Development <ul style="list-style-type: none"> <li>○ IR-framer</li> <li>○ USB-IR Bridge Controller</li> </ul> </li> <li>▪ System Integration</li> </ul>	<ul style="list-style-type: none"> <li>October 15, 2001</li> <li>October 15, 2001</li> <li>February 14, 2002</li> <li>February 28, 2002</li> <li>March 15, 2002</li> </ul>	<ul style="list-style-type: none"> <li>Hsiao</li> <li>Hou</li> <li>Hou</li> <li>Hou, Hsiao</li> <li>Hou, Hsiao, Hung</li> </ul>
<b>Final Implementation</b> <ul style="list-style-type: none"> <li>▪ Hardware and Software Integration</li> <li>▪ Testing and Debugging</li> </ul>	<ul style="list-style-type: none"> <li>March 22, 2002</li> <li>March 31, 2002</li> </ul>	<ul style="list-style-type: none"> <li>Hou, Hsiao, Hung</li> <li>Hou, Hsiao, Hung</li> </ul>

Documentation and Presentation		
▪ Interim Report	January 11, 2002	Hou, Hsiao, Hung
▪ Oral Presentation	February 14, 2002	Hou, Hsiao, Hung
▪ Poster Presentation (Design Fair)	March 18-20, 2002	Hou, Hsiao, Hung
▪ Final Report	April 12, 2002	Hou, Hsiao, Hung

## ***G Team Member Contribution***

*Eric Hsiao*

I composed Chapter 3 USB Module, Chapter 4 USB-IR Bridge and the Appendices of the final report.

I designed the following components of the project:

- Hardware of the USB module
- VHDL codes of the USB-IR bridge
- Schematics of the USB-IR bridge
- Simulation scheme of the USB-IR bridge
- Layout of the USB module and USB-IR bridge

The USB-IR bridge design includes:

- Modulation Scheme
- Demodulation Scheme
- Data Direction Controllers
- Samplers

*Richard Hou*

I have written Chapter 5, Infrared Module in this final report.

During the process of design process, I had dedicated all my time and efforts to the research of Infrared technology, design of transmitter and receiver circuits, acquiring and testing all components which might be suitable for the design, and implementing and testing every stage of transmitter and receiver circuits on breadboard and vector board.

Infrared module is consisted of transmitter and receiver. The main component of transmitter is a current driver circuit. Receiver has four building blocks: Photodiode, Preamplifier, Comparator, and Threshold Level Detector.

I was also involved in discussions and development of USB-IR bridge module in the beginning stage.

*Roger Hung*

Throughout the term, I have investigated the USB signaling and data transfer characteristics and helped establishing USB-IR communication protocol. I was mainly responsible for implementation and it concludes constructing PCB and helping to make the test board. Also, I participated in development on USB-IR module in the beginning stage and device testing of whole system.

For this report, I wrote the Introduction, Background Information, System Implementation and Conclusion.